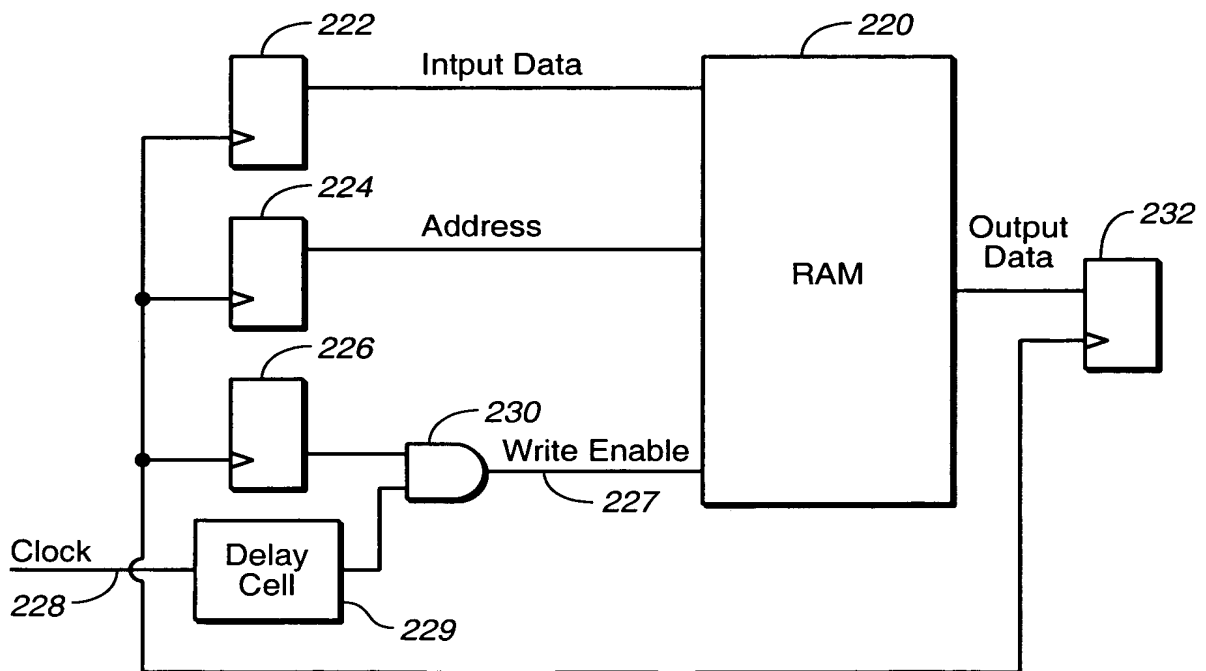
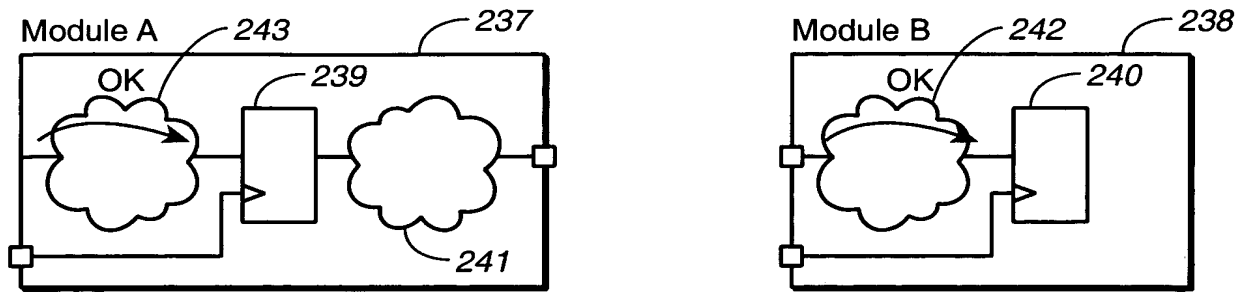
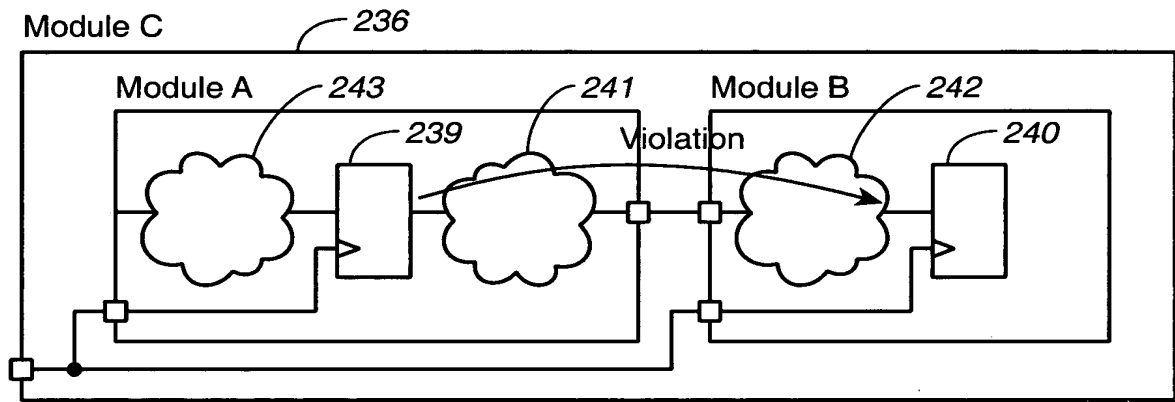
**FIG._4****FIG._5**

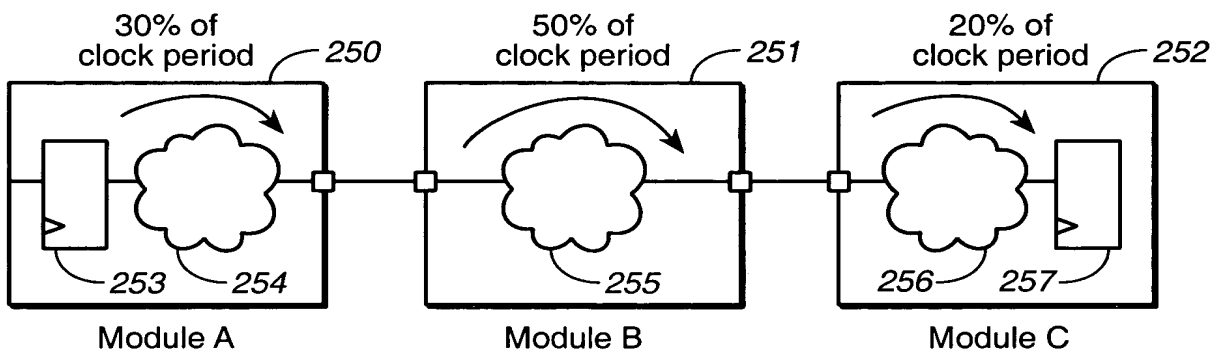


1. Exploratory synthesis on individual modules meets clock constraints.



2. Violating paths show up when modules are put together

FIG._6



Modules A,B and C compiled independently

FIG._7



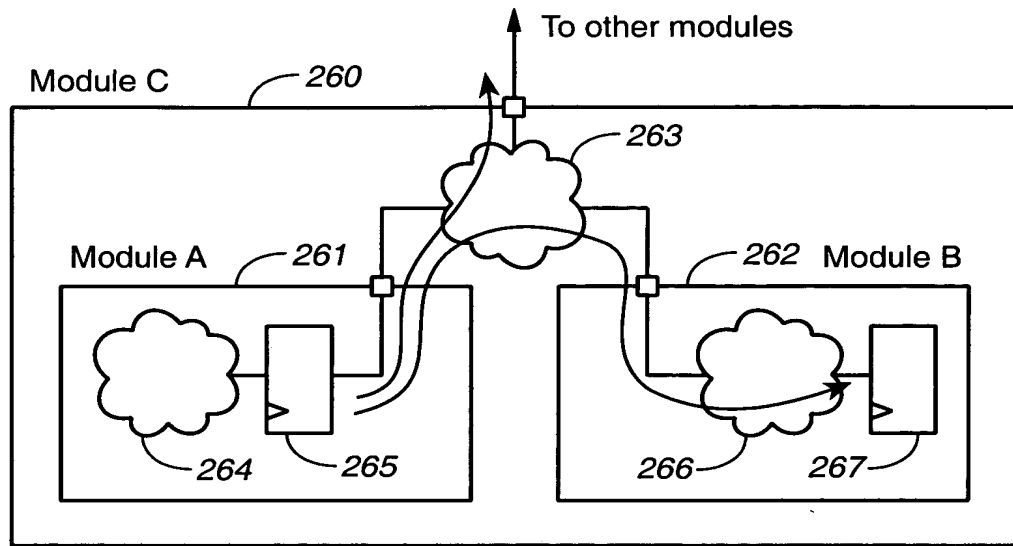
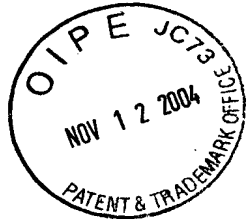
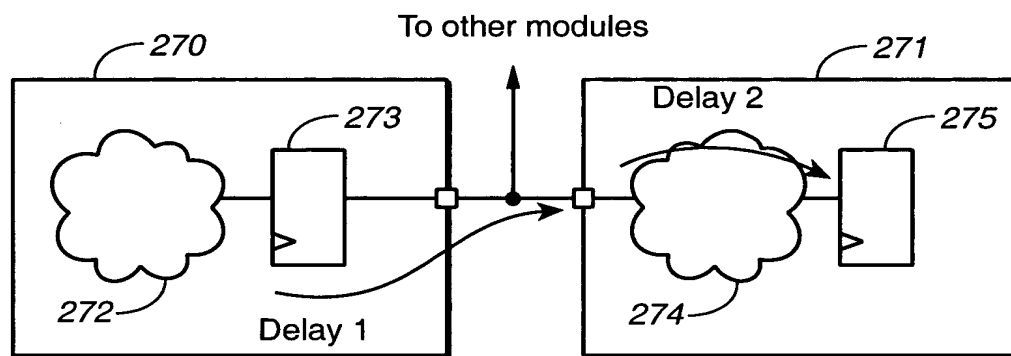


FIG. 8



Delay 1 << Delay 2

FIG. 9

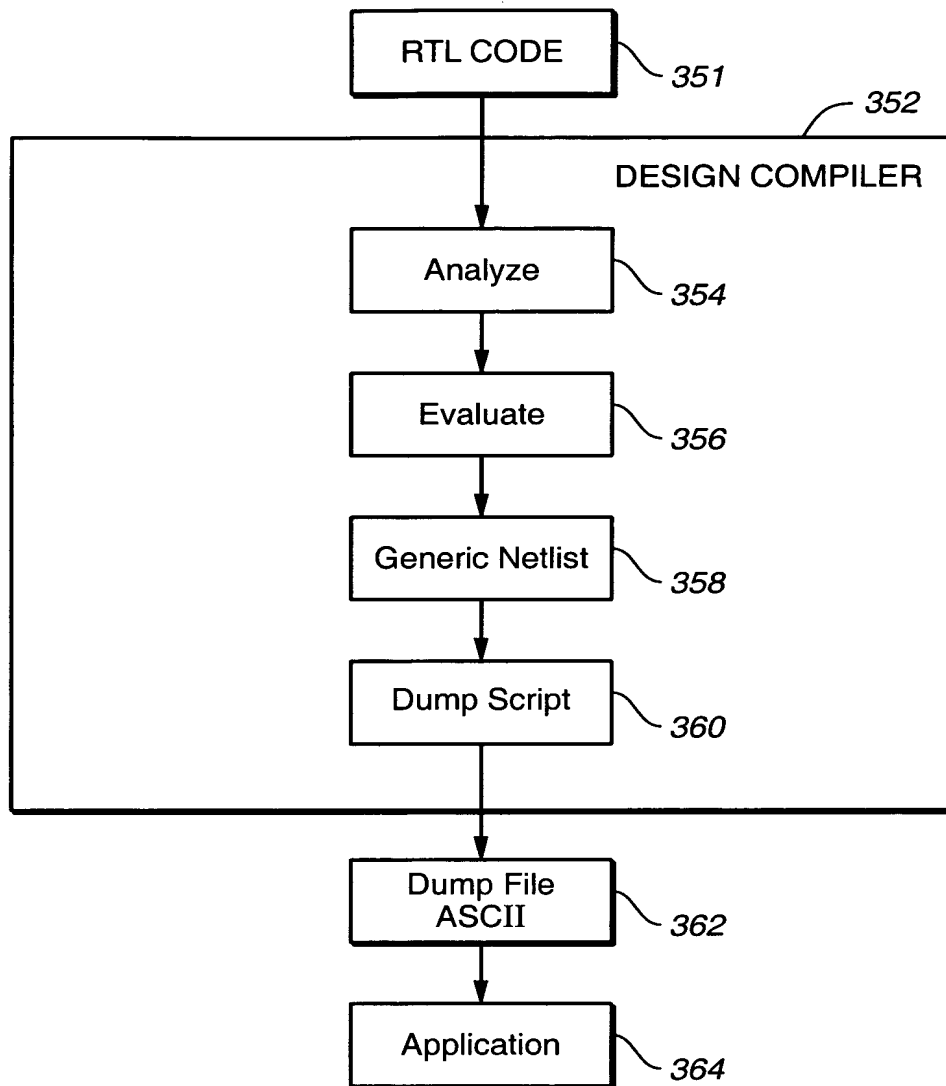


The diagram illustrates a scan chain architecture for a RAM component. A central block labeled "RAM" is connected to a scan chain. The scan chain consists of several components: a flip-flop (286), a cloud-shaped logic block (288), a multiplexer (290), the RAM block (280), another multiplexer (291), another cloud-shaped logic block (289), and a final flip-flop (287). A "Scan Enable" signal is connected to the multiplexers (290 and 291) and the RAM block (280). A "Specific logic added to isolate RAM in scan mode" is shown as a block connected to the RAM block (280) and the multiplexer (291). A "Scan Chain" label points to the overall sequence of components. A "Combinational logic in RAM's shadow" label points to the cloud-shaped logic blocks (288 and 289). A label "292" points to the connection between the multiplexer (290) and the RAM block (280).

With combinational logic in RAM's shadow

FIG. 10B



**FIG._12**

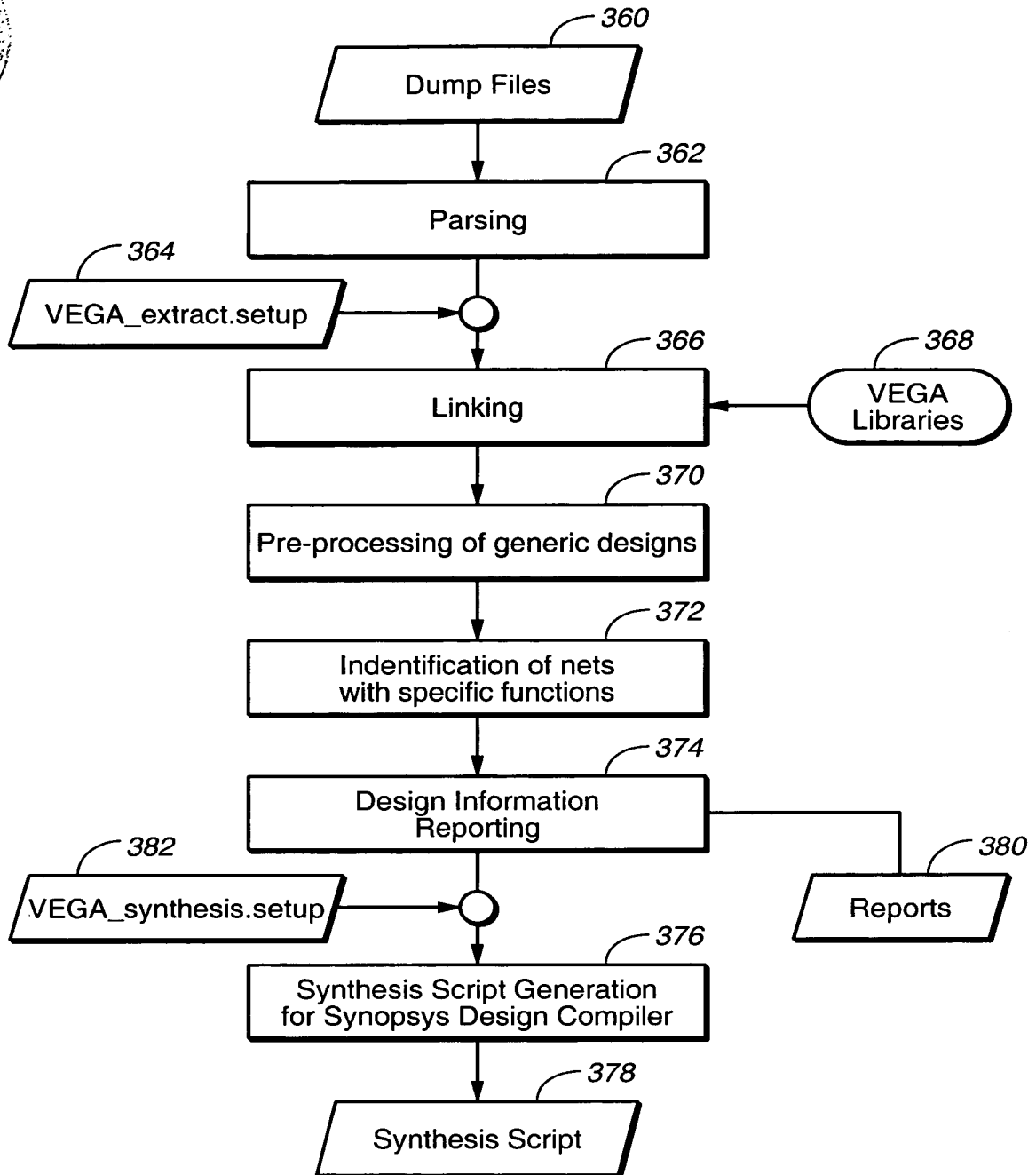
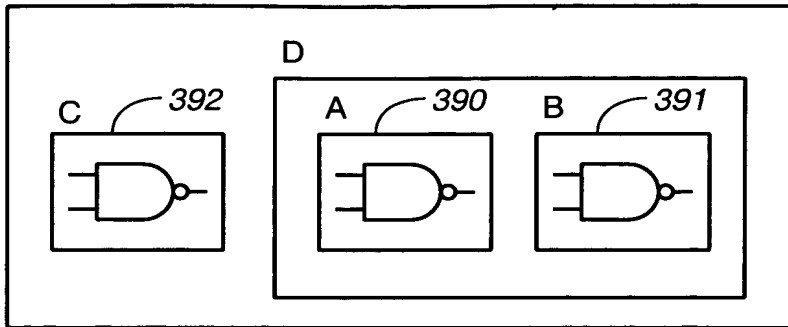


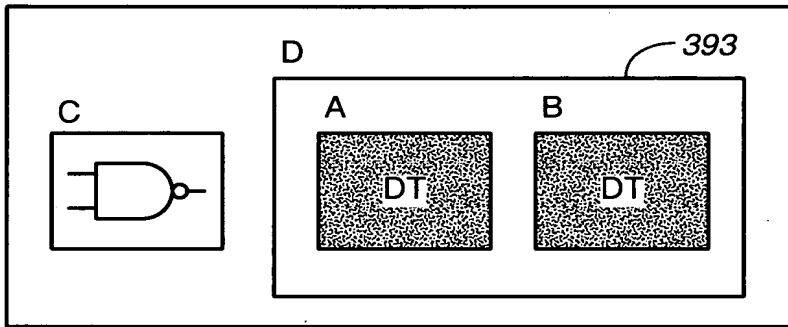
FIG. 13
VEGA FLOW

Top

Step #1

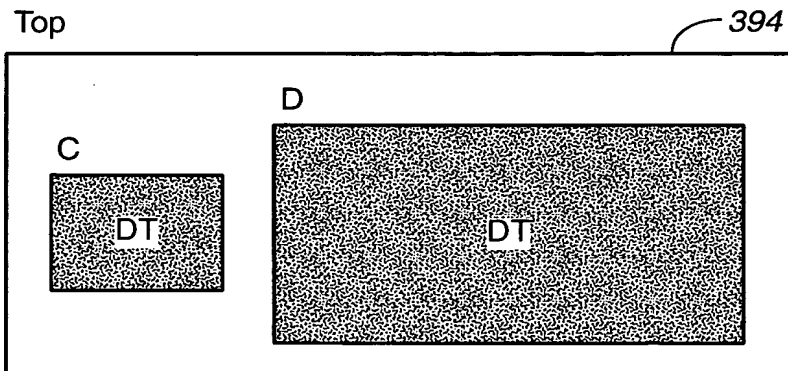
: Leaf modules A, B and C are synthesized.

Top

Step #2

: Module D is synthesized with modules A and B made non-modifiable (don't-touch attributes).

Top

Step #3

: Module TOP is synthesized with modules C and D made non-modifiable.

FIG. 14 : Bottom-up synthesis.



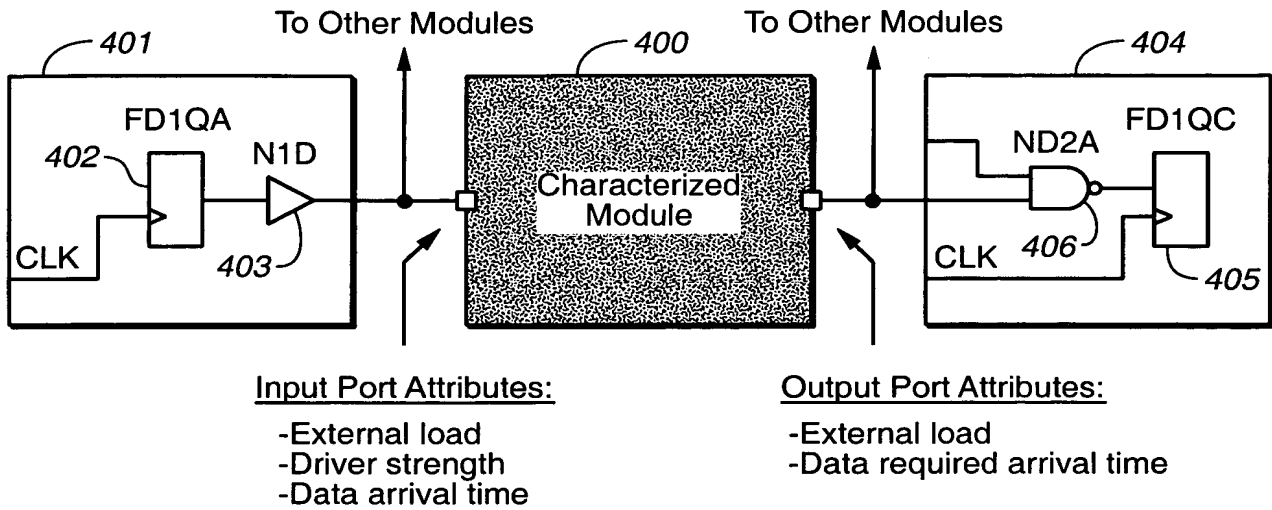


FIG._15 :Characterization.

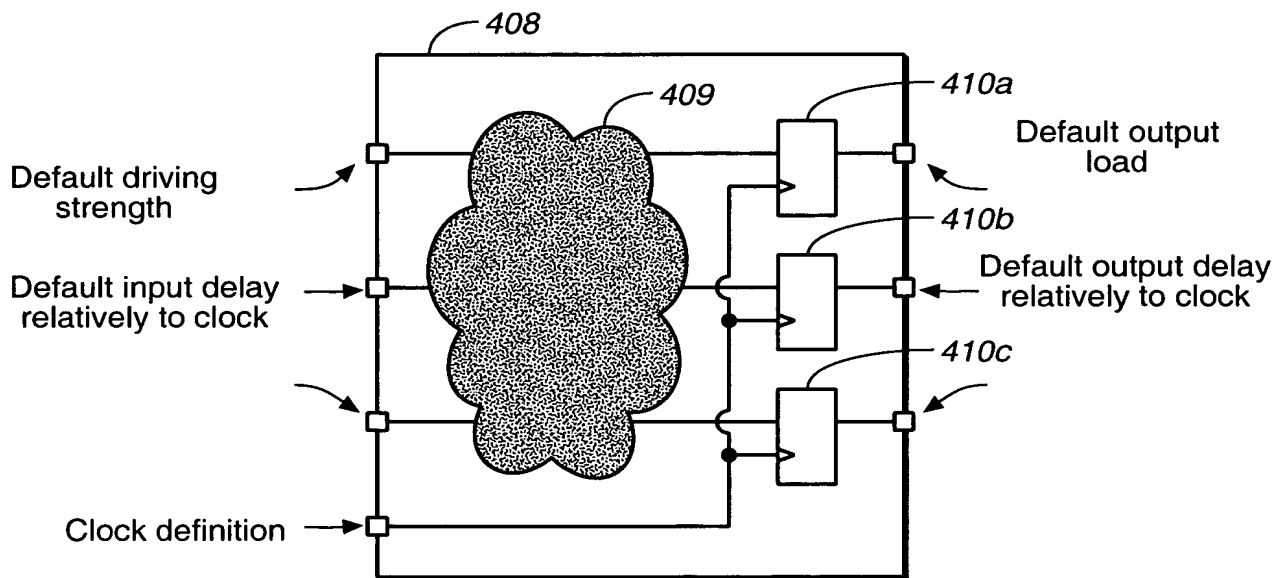


FIG._17 :Default constraints used for initial mapping.

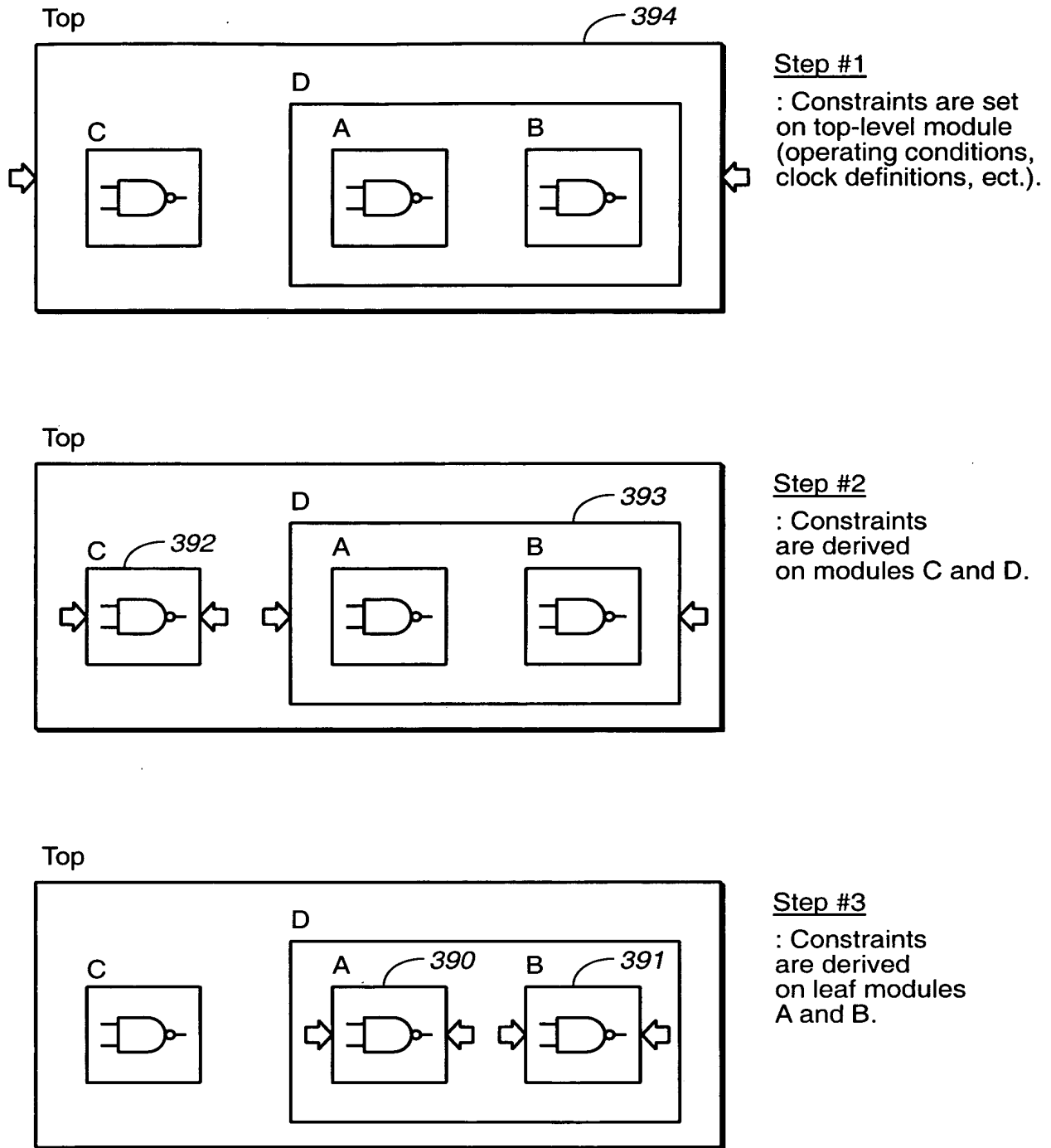


FIG. 16 : Top-down characterization.

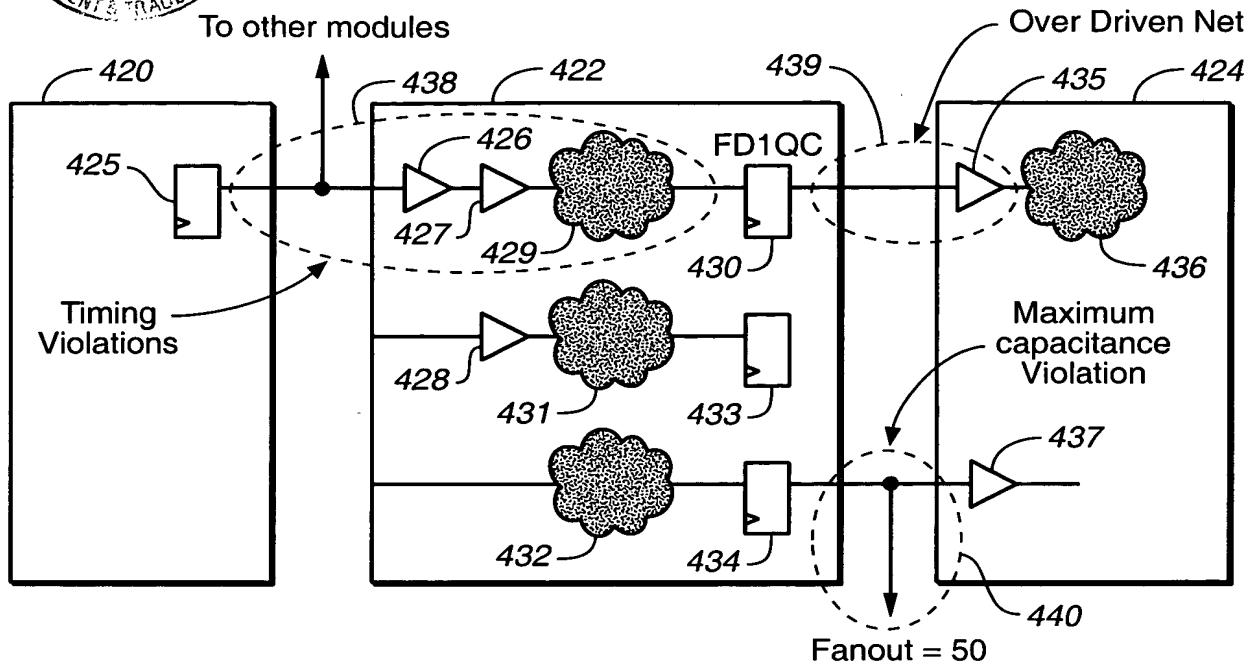
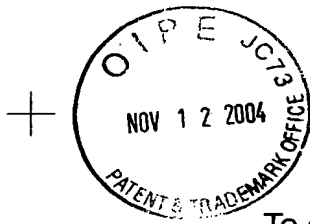


FIG. 18 : Results after initial mapping.

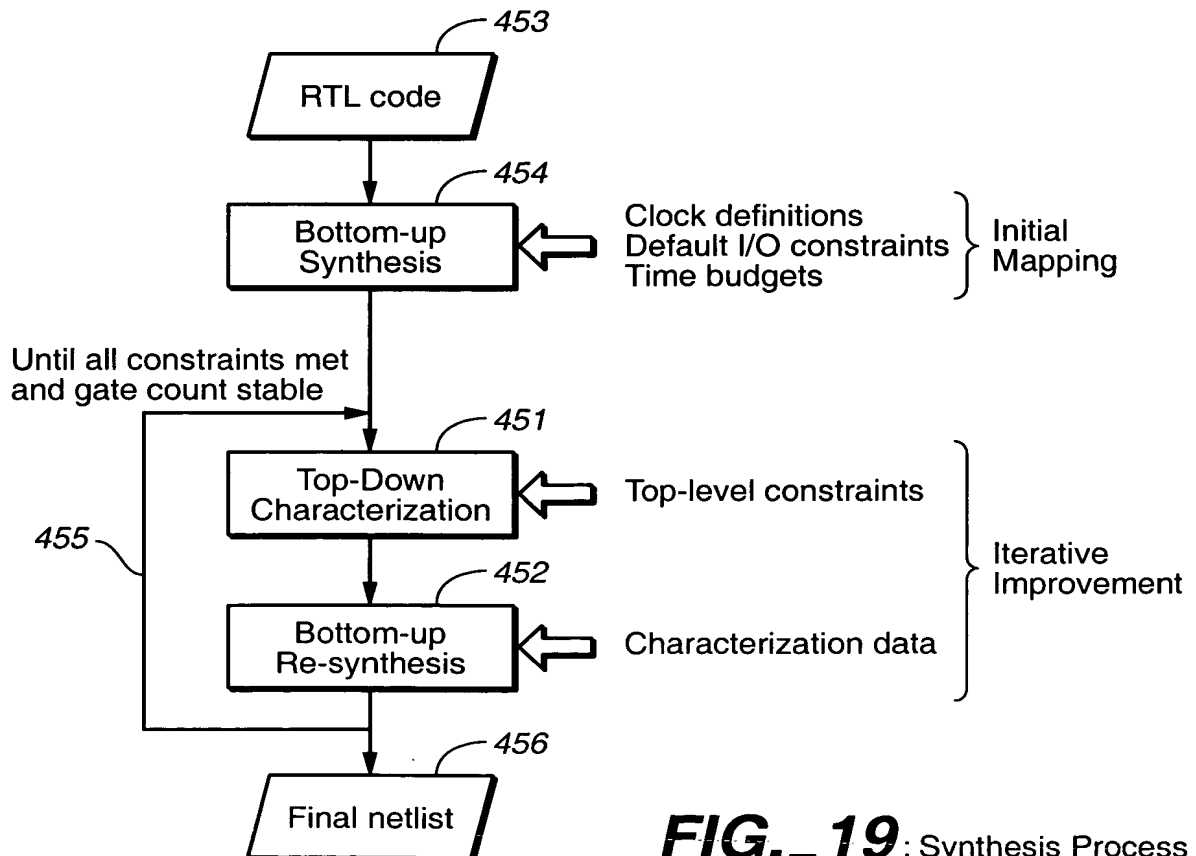


FIG. 19 : Synthesis Process

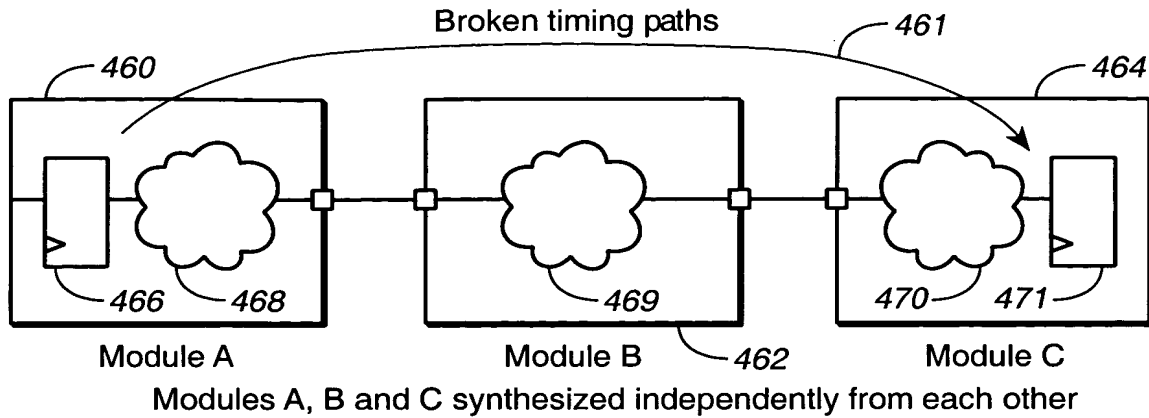


FIG._20A : Broken timing paths

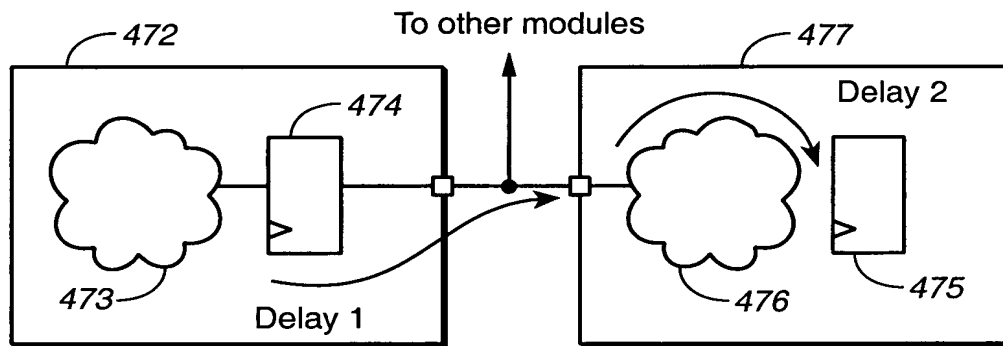


FIG._20B : In the absence of broken timing paths

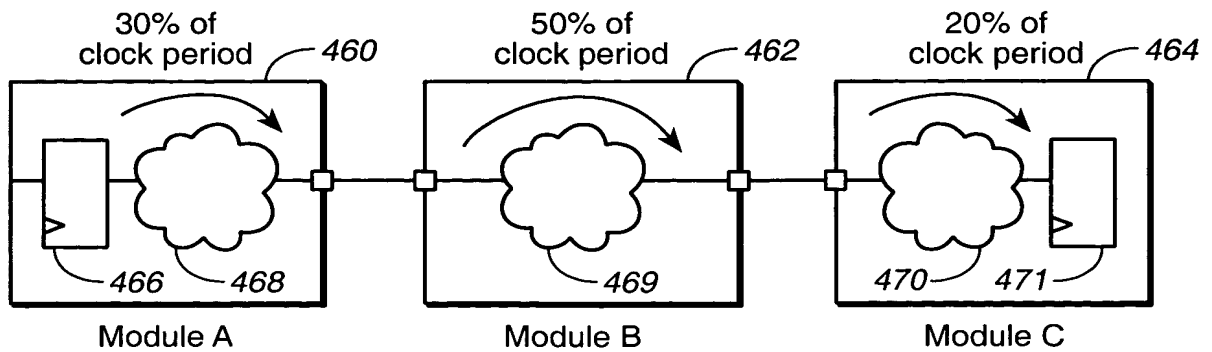


FIG._20C : Time budgets.

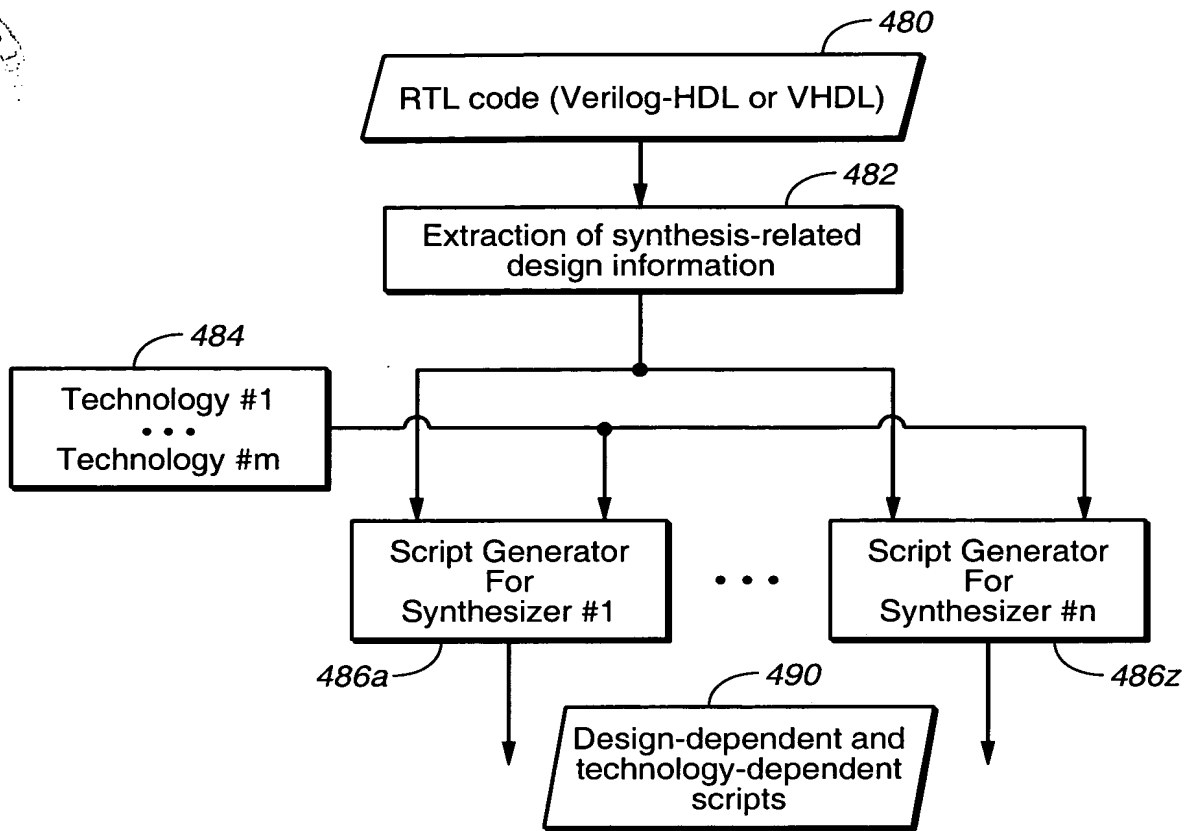
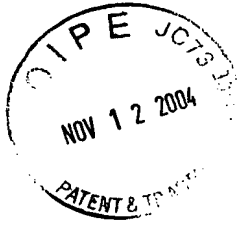
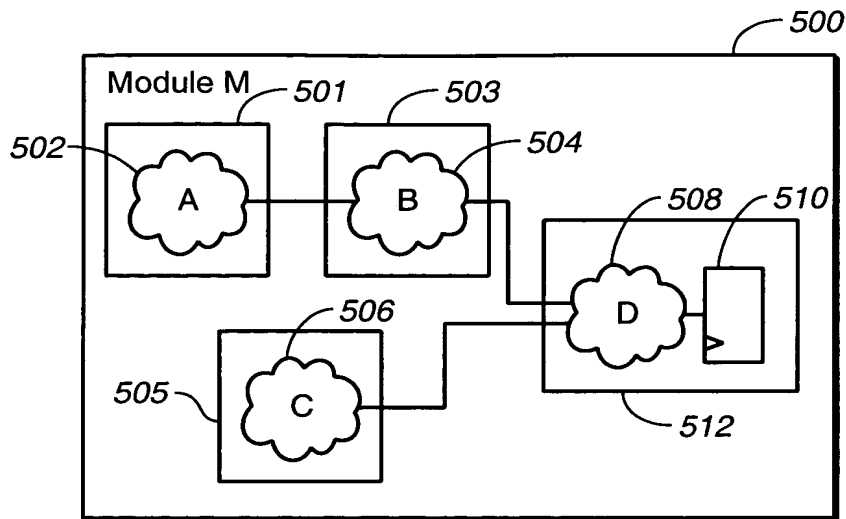
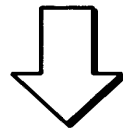


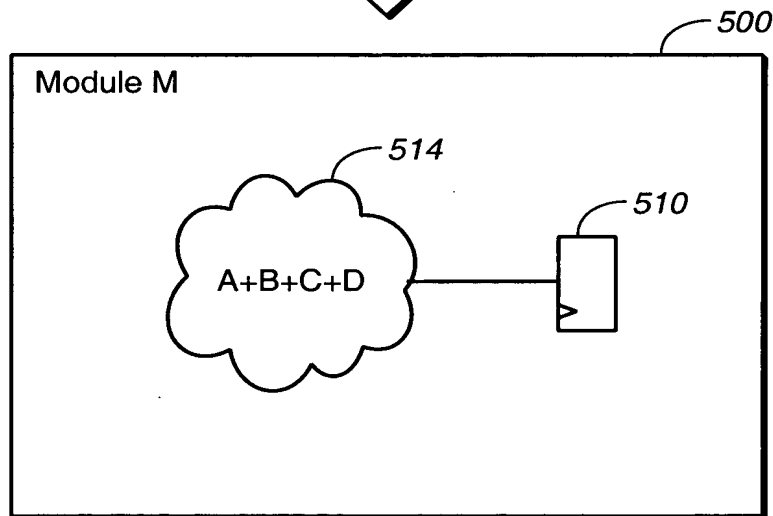
FIG._21 : Automatic Script Generation



logic clouds A, B, C and D cannot get combined together, because they are embedded in different modules.

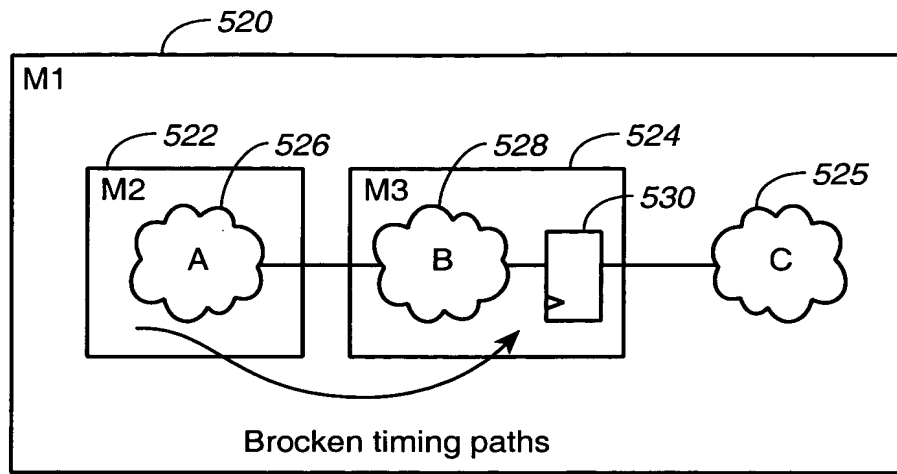


Ungrouping



logic clouds A, B, C and D can get combined after dissolving the design hierarchy under module M.

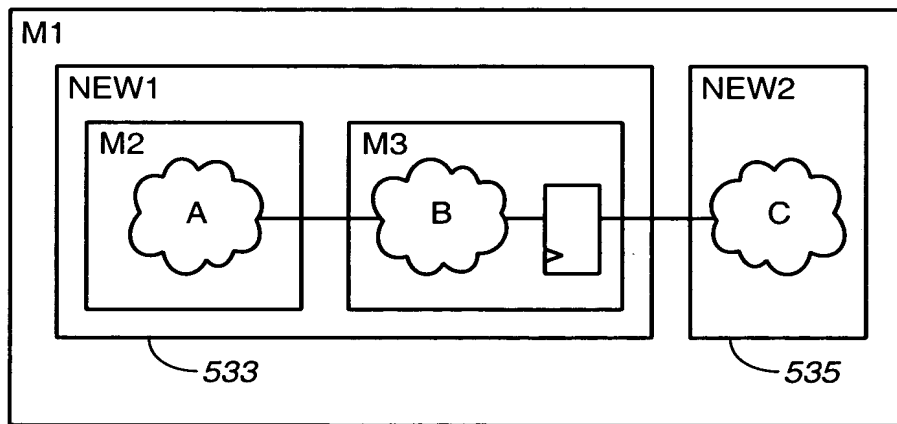
FIG._22: Hierarchy ungrouping.



Broken timing paths run through modules M2 and M3. Module M1 mixes hierarchy (modules M2 and M3) with logic (cloud C).



Grouping



M2-M3 timing paths are now fully contained in new modules NEW1. If modules M2 and M3 are small enough, the hierarchy can be dissolved below NEW1. New module NEW2 encapsulates cloud of logic C.

FIG. 23: Hierarchy grouping.

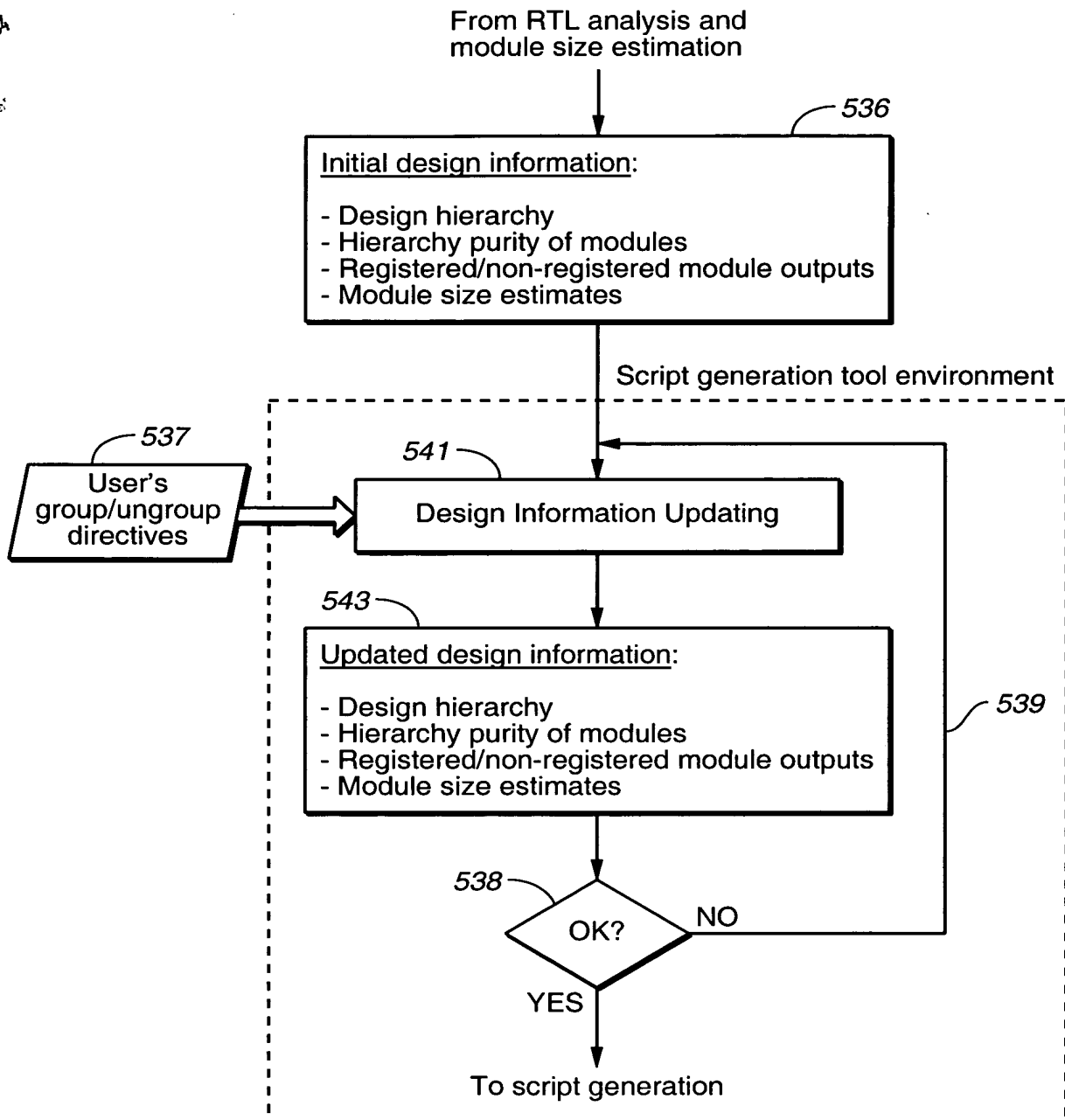
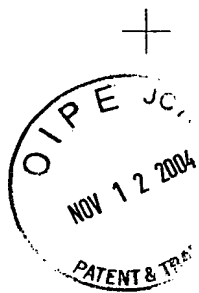


FIG._24 : Support for design hierarchy re-arrangement

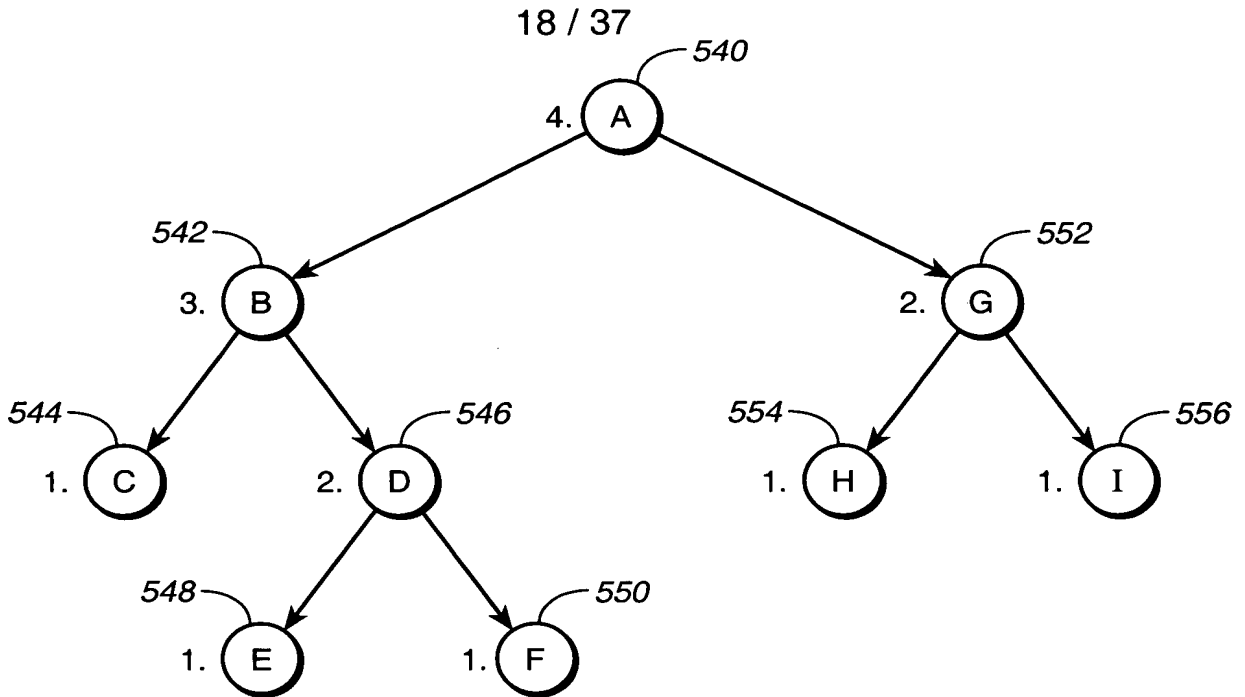


FIG._25 : Module processing order for parallel bottom-up synthesis.

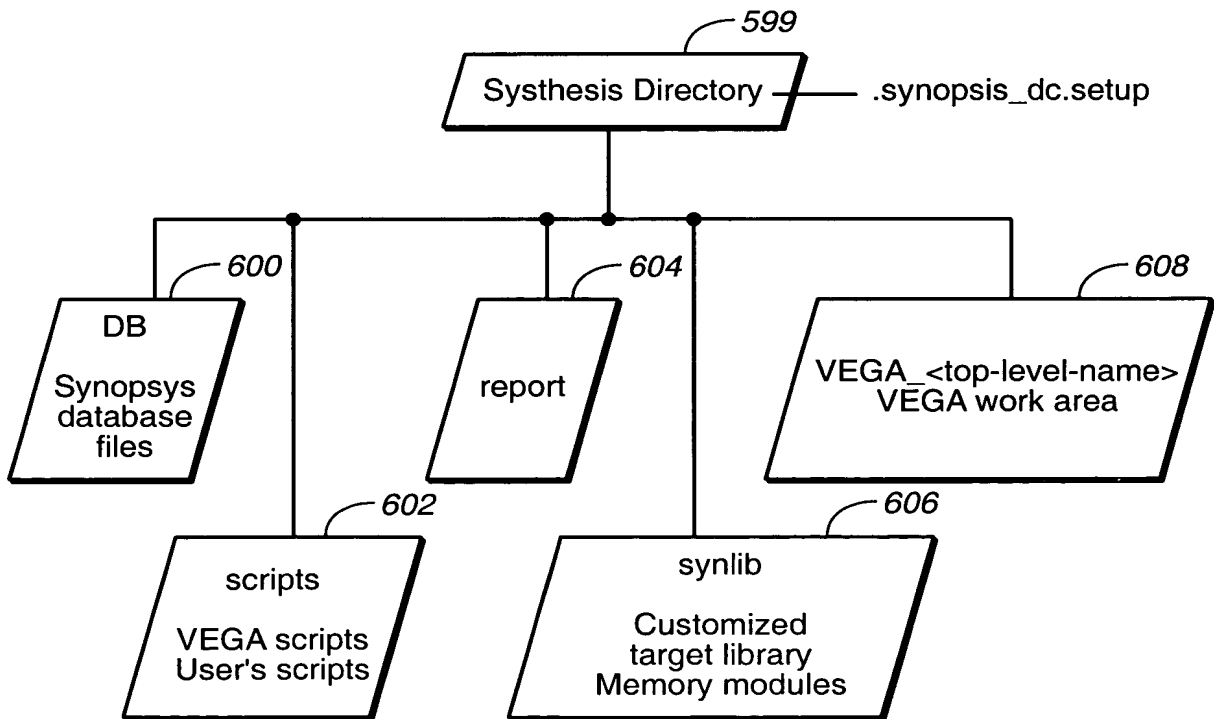


FIG._26 : Database to be used to run VEGA scripts.

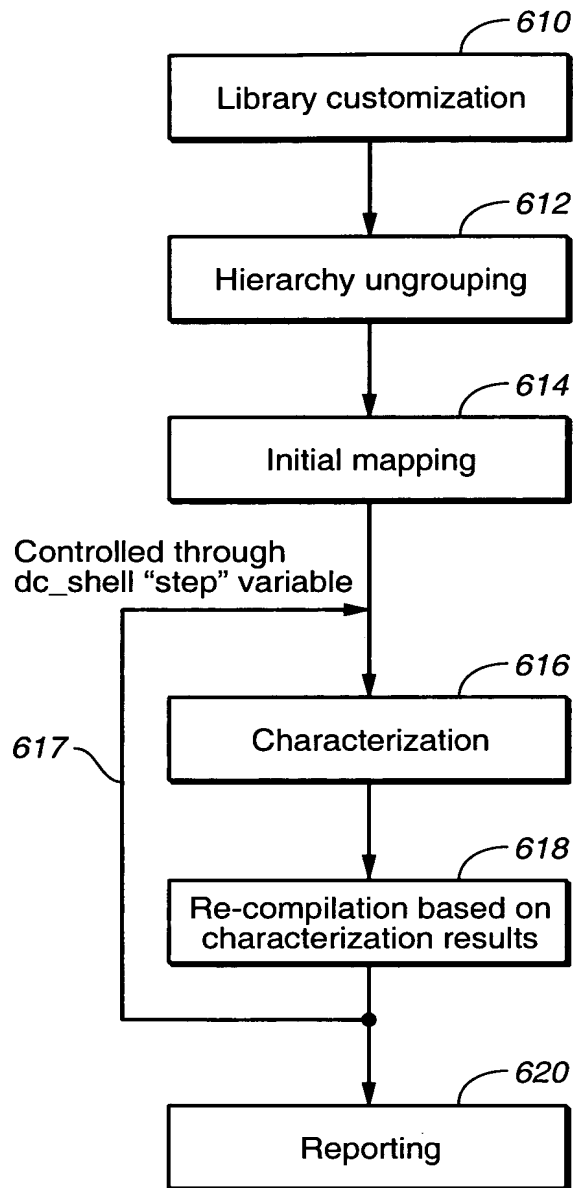
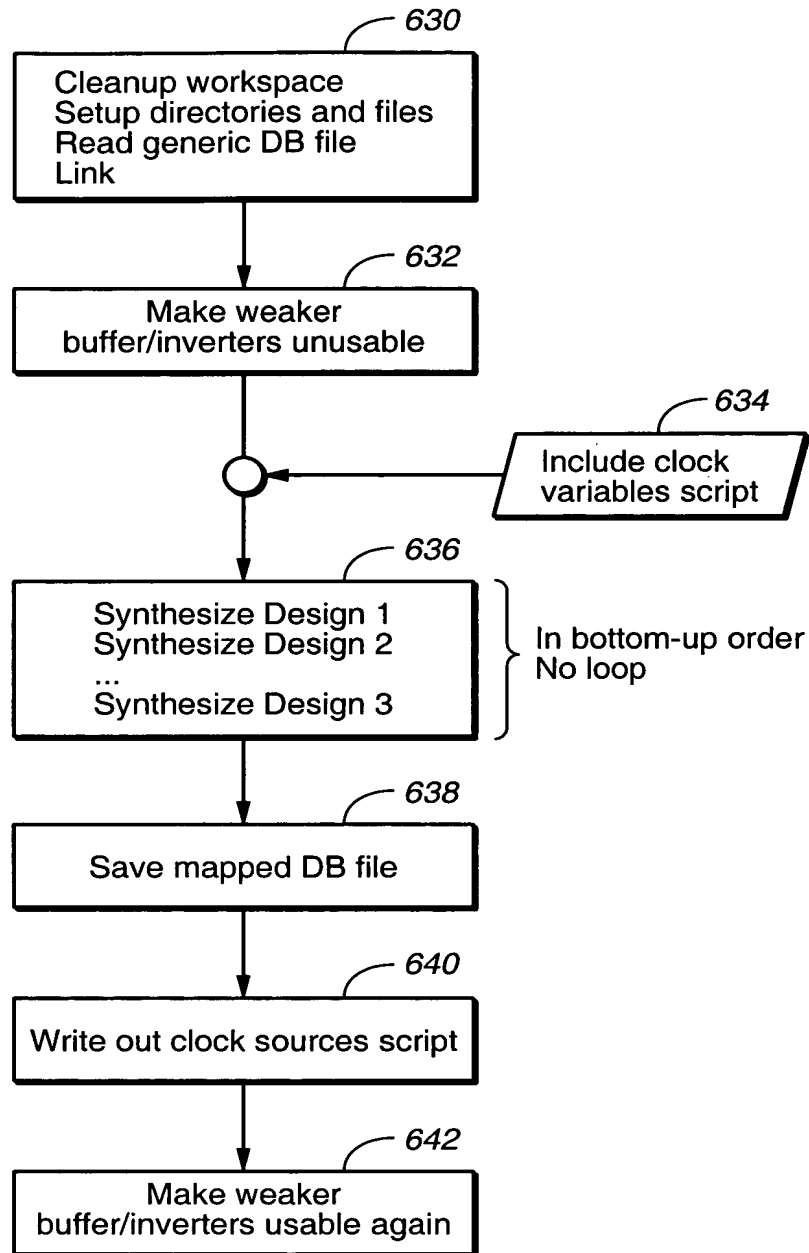
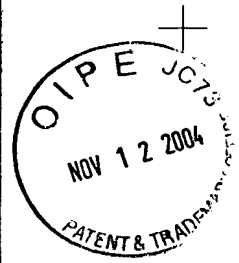
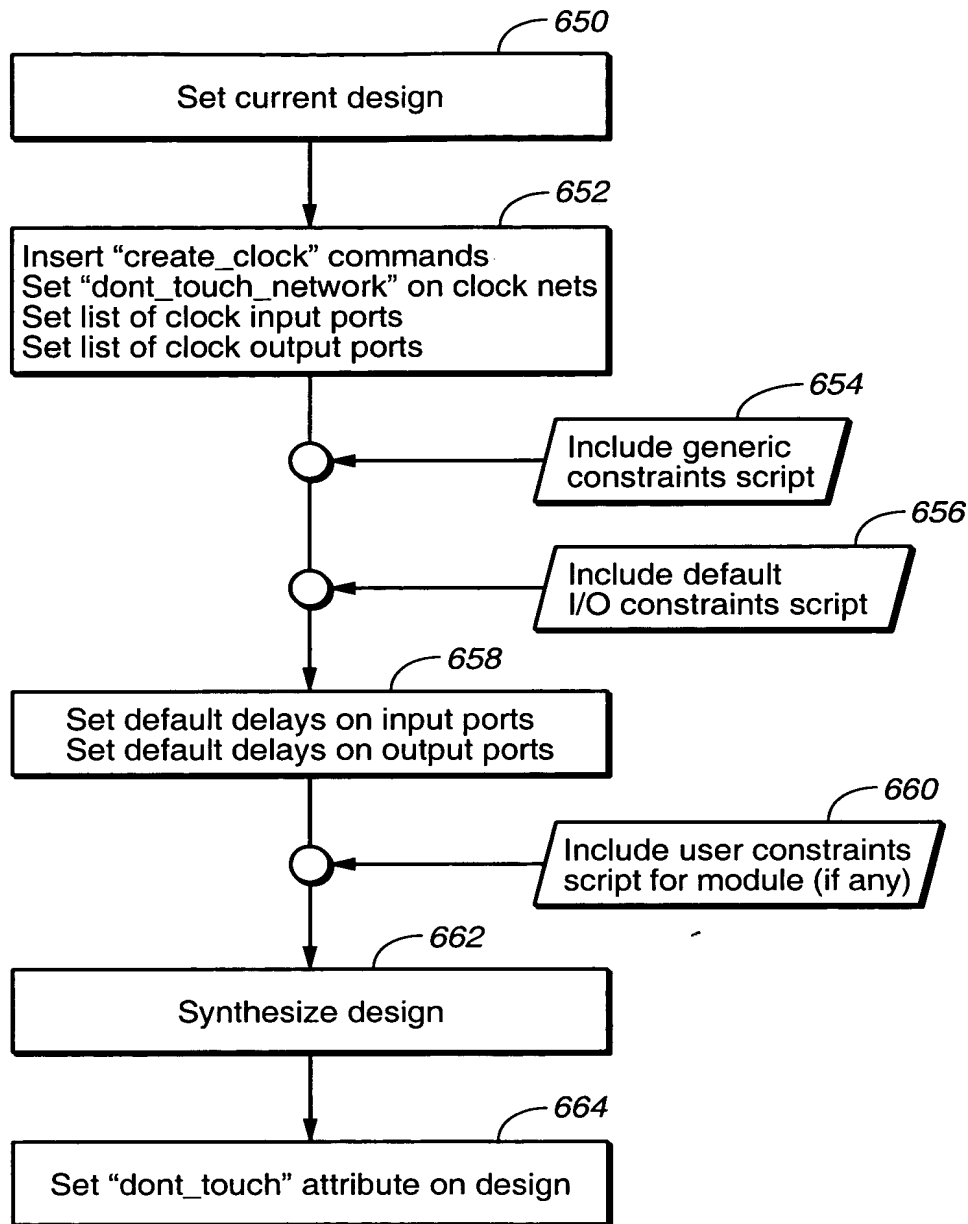
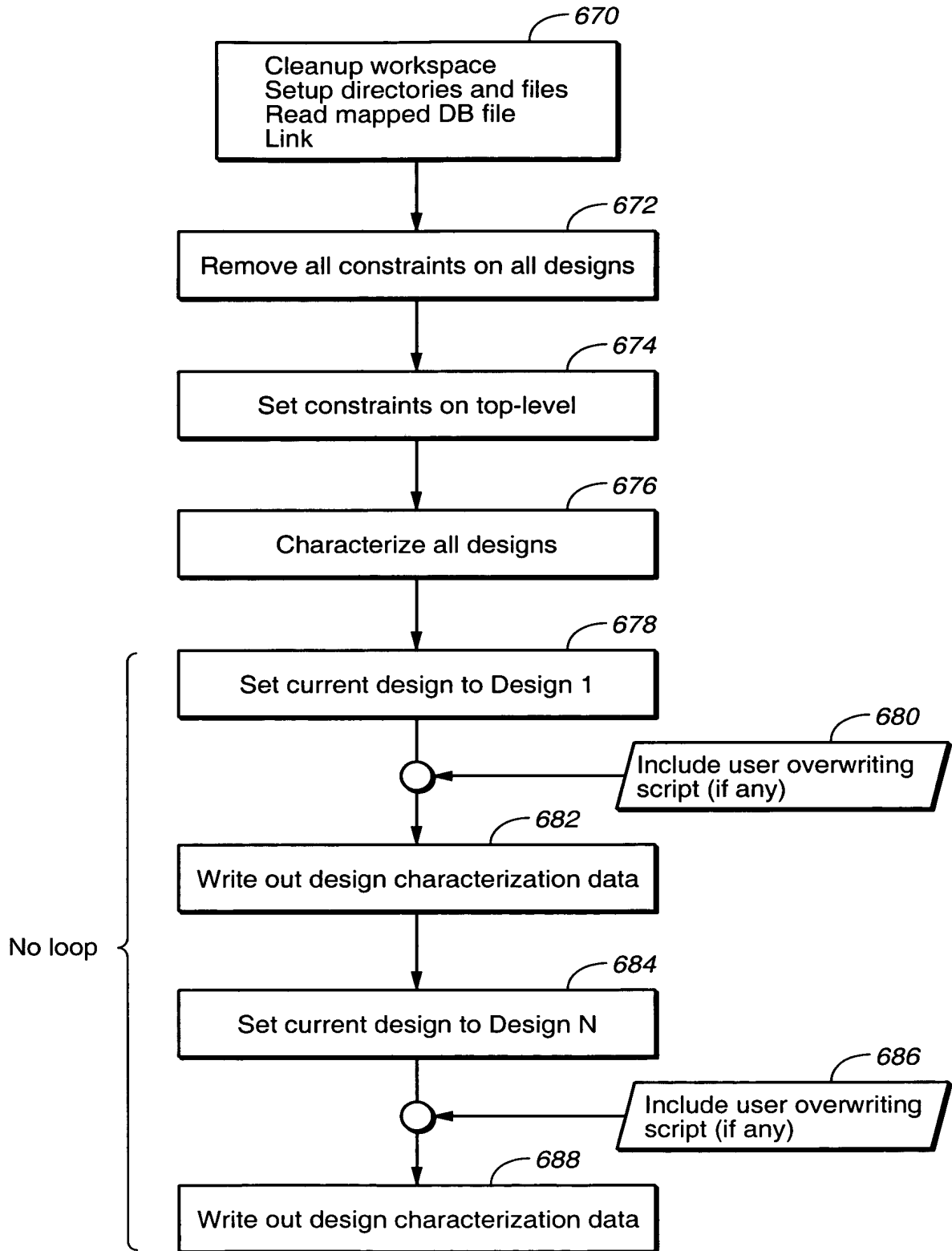


FIG._27 : Script flow implemented by VEGA

**FIG._28** : Structure of initial mapping script

**FIG. 29**

: Operations performed on each module by initial mapping

**FIG._30** : Structure of characterization script

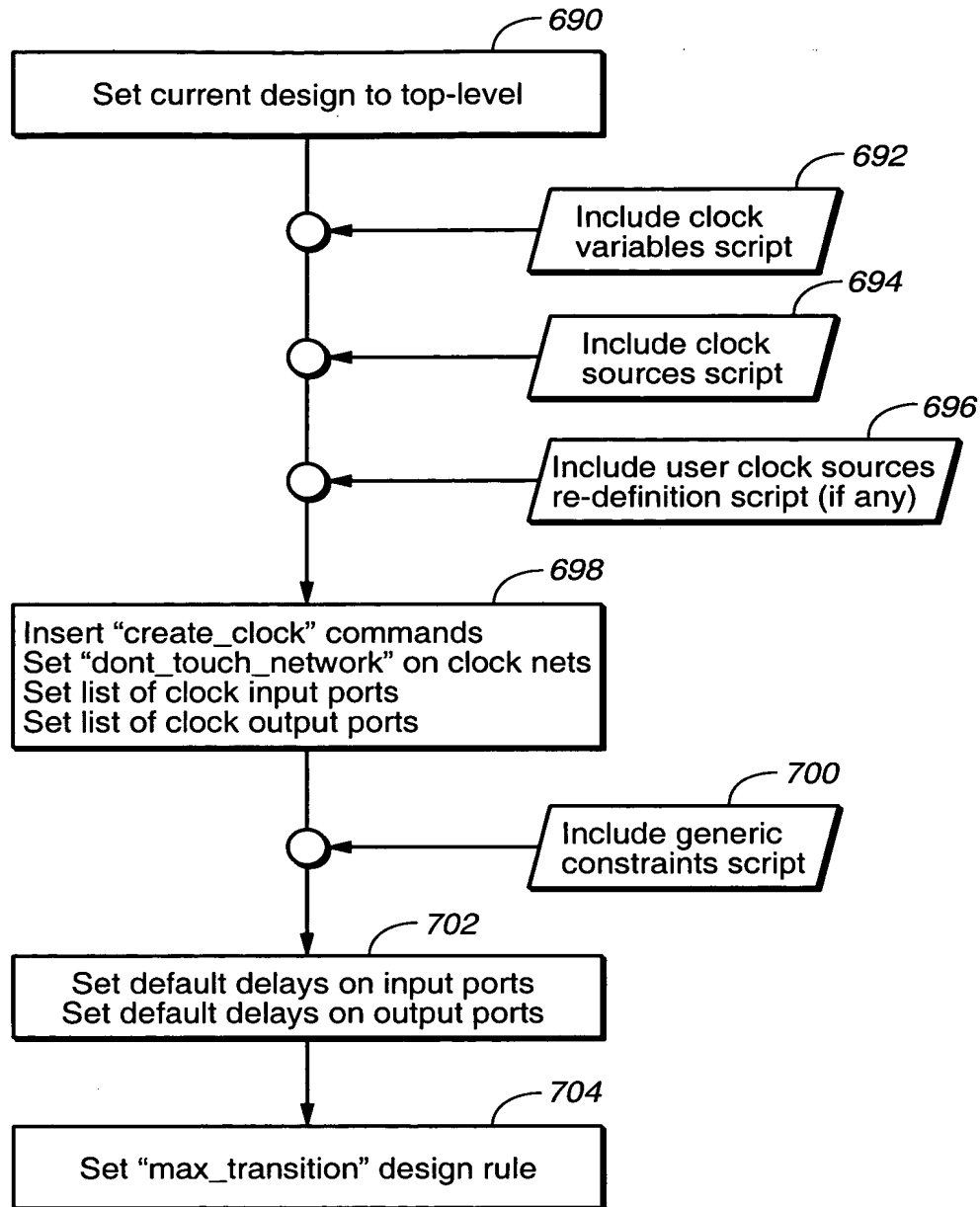
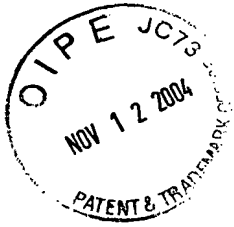


FIG._31 : Structure of constraints setting on top-level

24 / 37

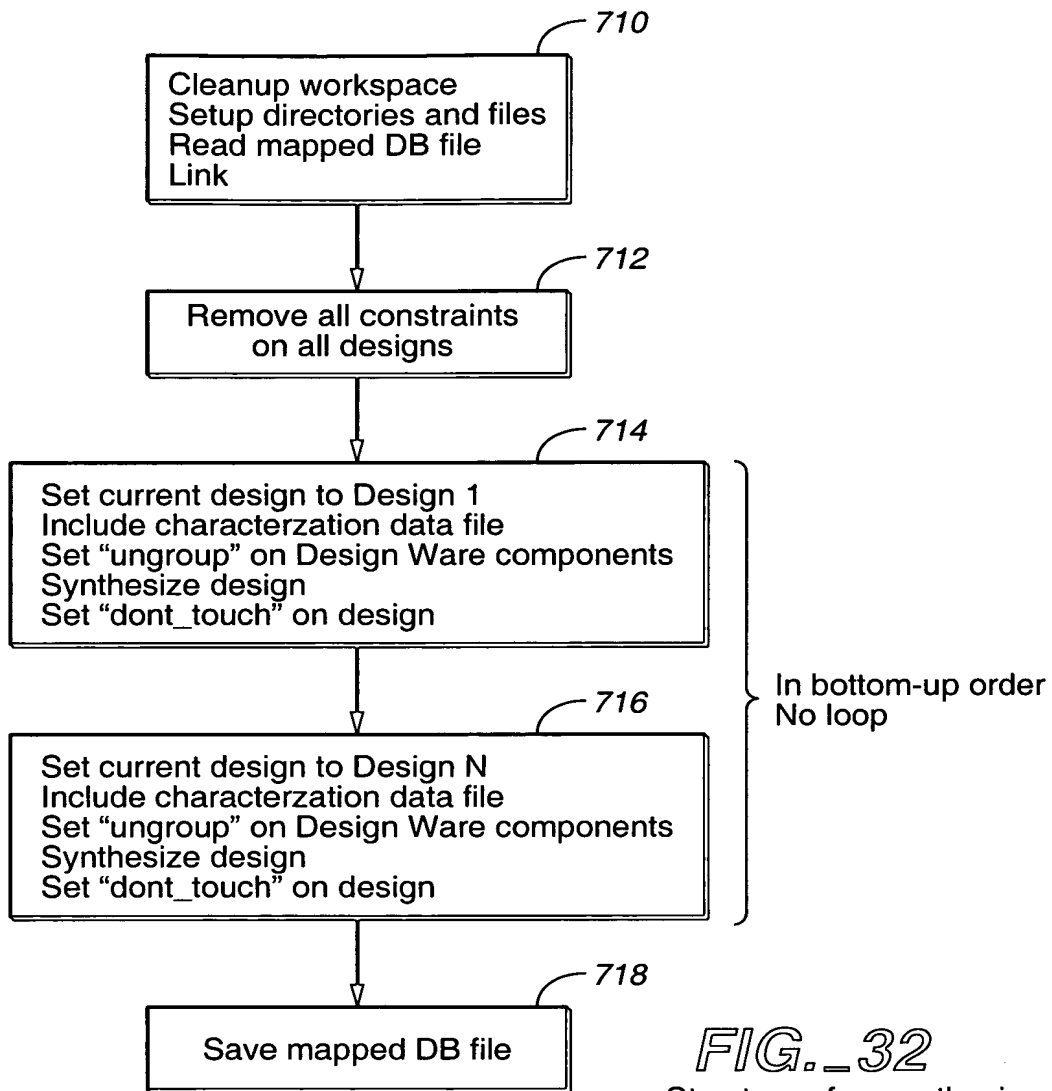


FIG. 32
: Structure of re-synthesis script

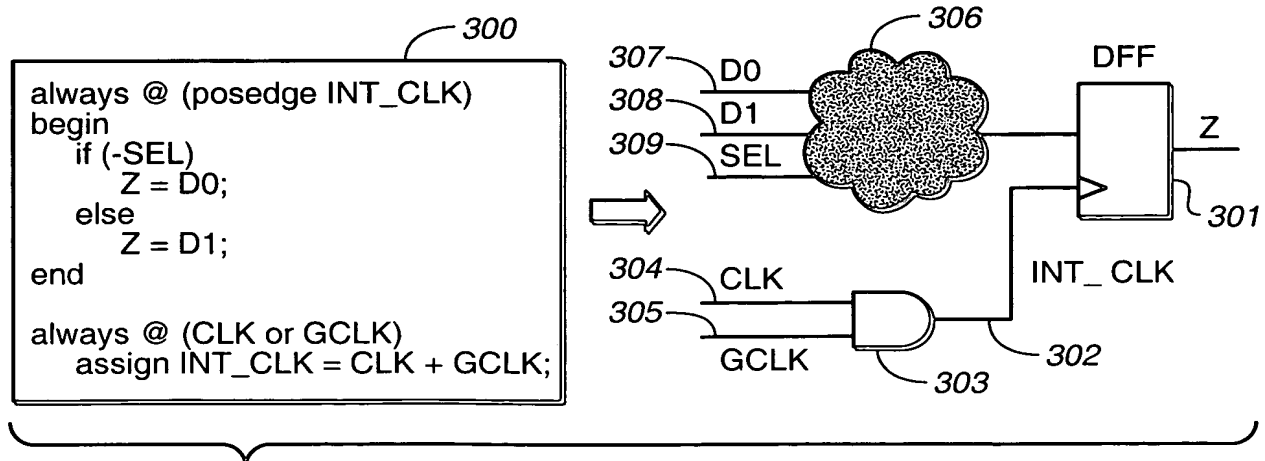
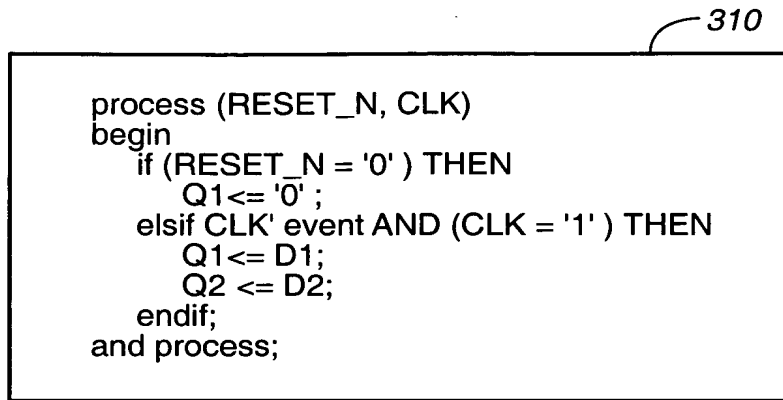
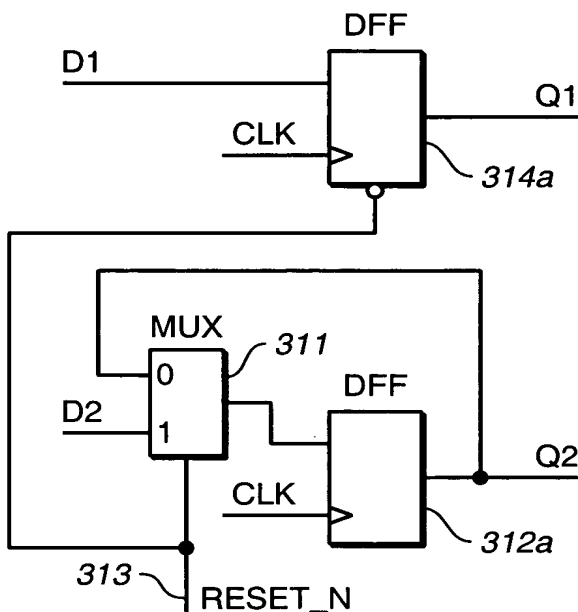
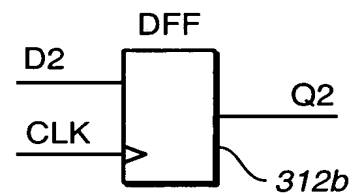
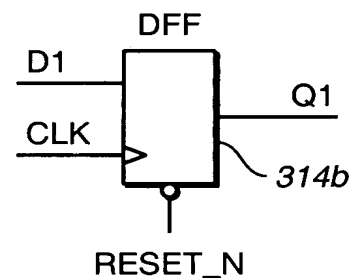


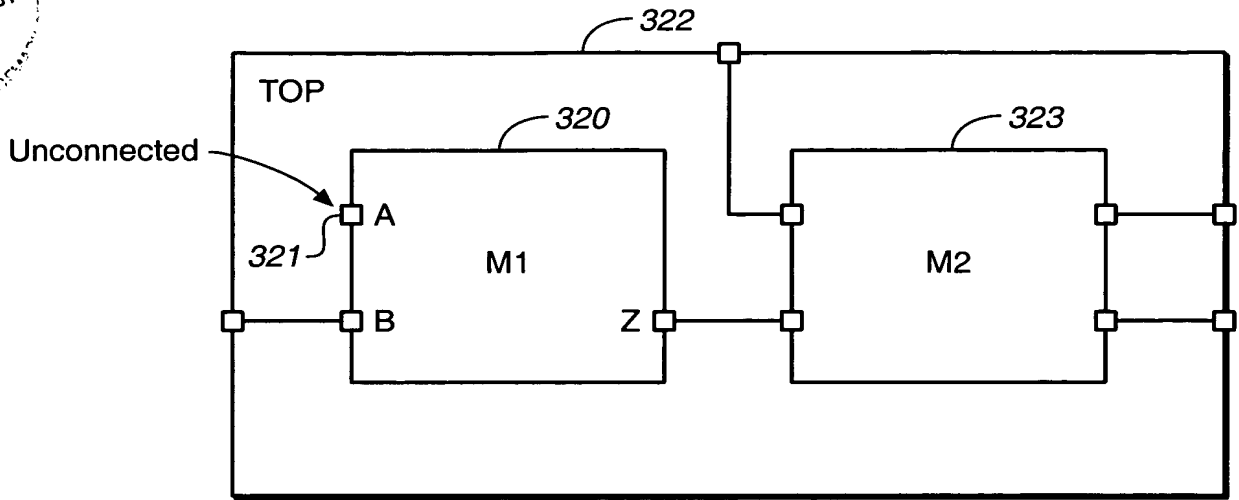
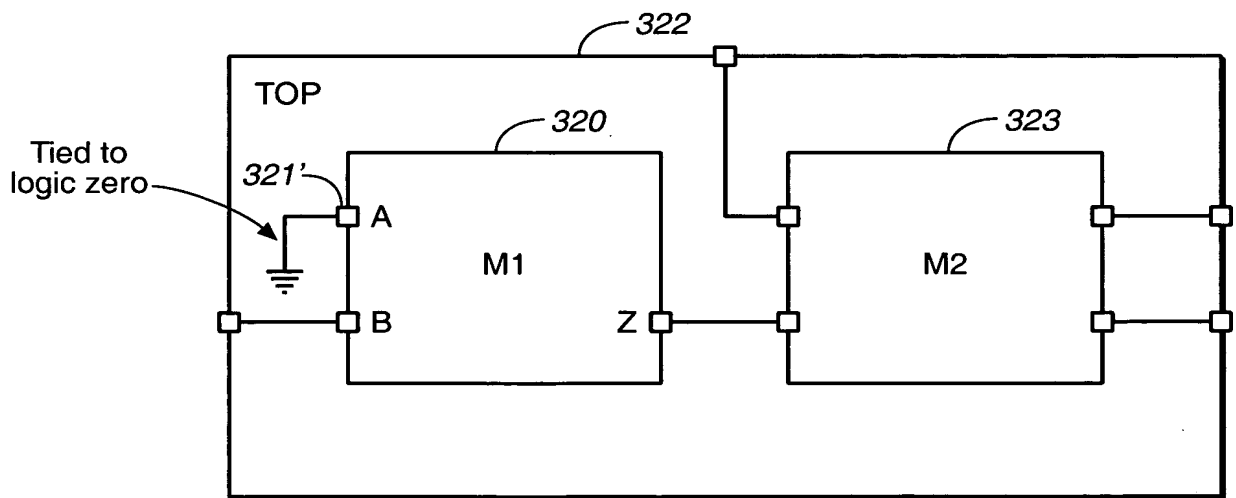
FIG. 33 : Example of RTL code and equivalent.
Hardware view for RTL analysis.



a. VHDL code for a 2-bit register with partial asynchronous reset

FIG._34Ab. Implementation created by
Synopsys Design Compiler**FIG._34B**c. Implementation created by
AMBIT BuildGates**FIG._34C**

Implementation of partial asynchronous reset.

**FIG._35A** : RTL code**FIG._35B** : Synopsys Design Compiler view of the RTL code

Handling of unconnected module input pins by Synopsys Design Compiler.

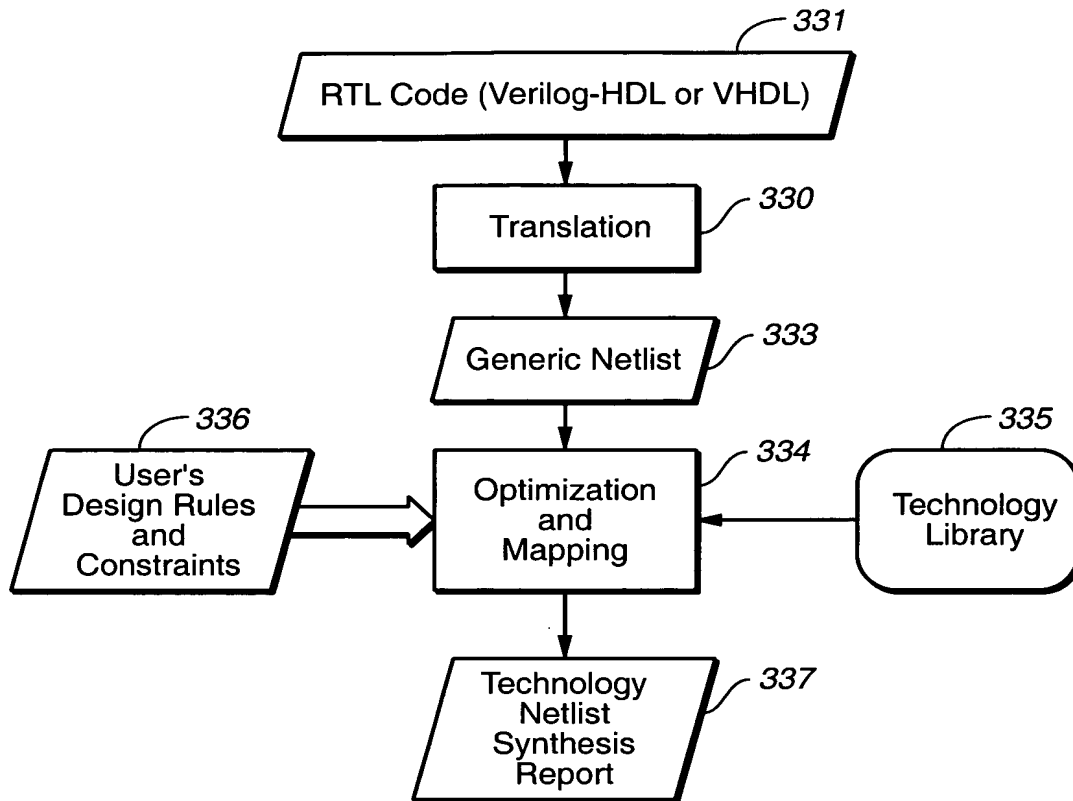


FIG._36 : Logic synthesis process

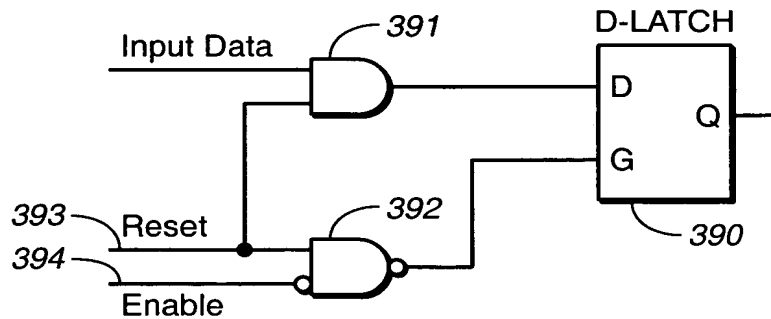
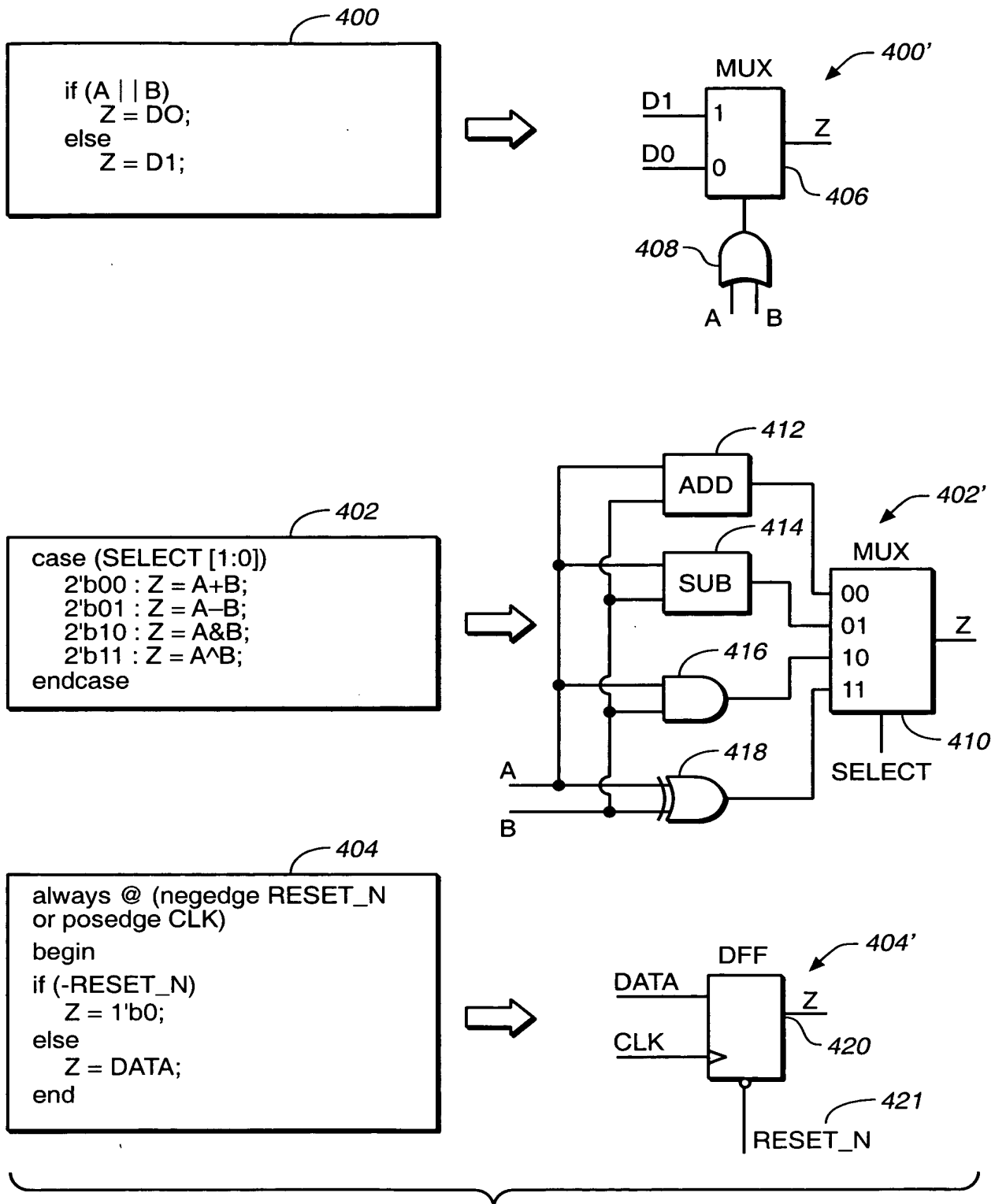


FIG._37 : Failing implementation of a latch with clear.

28 / 37

**FIG. 38**

: Examples of transforms used for RTL code translation

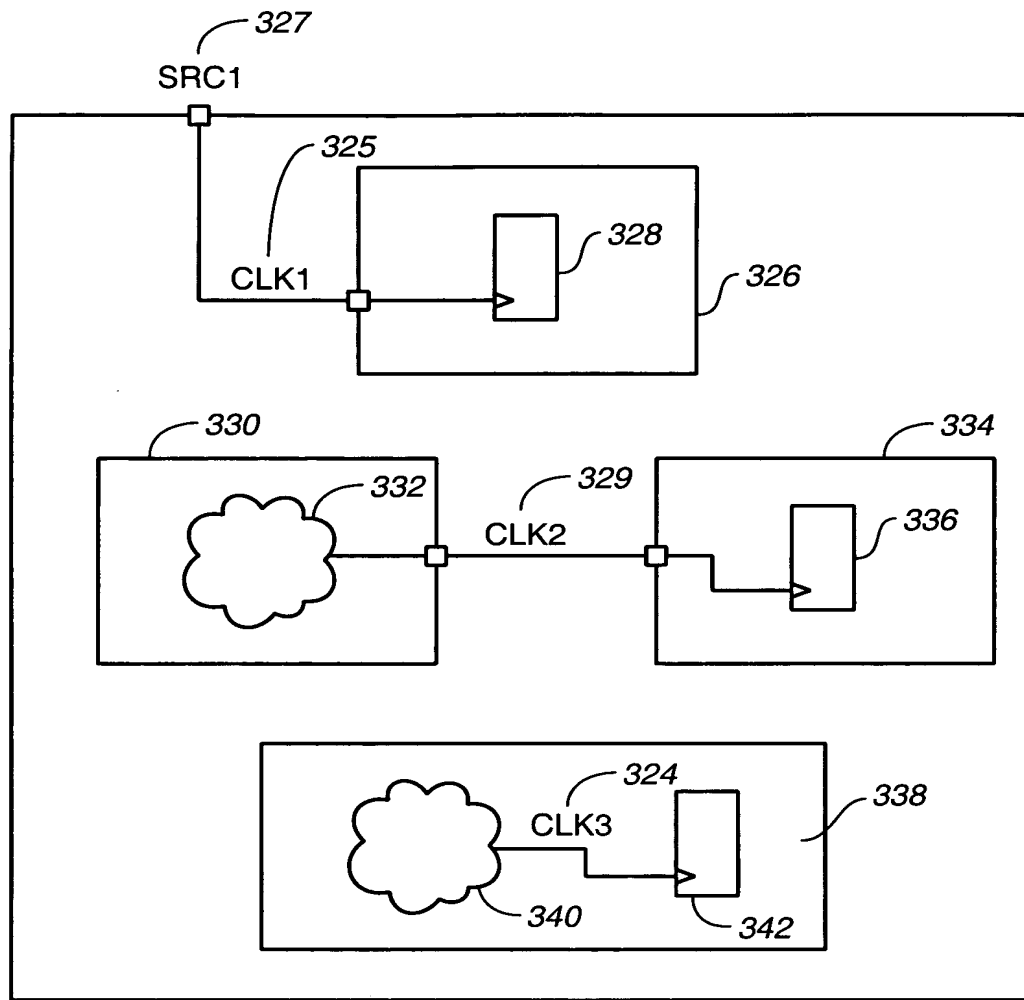
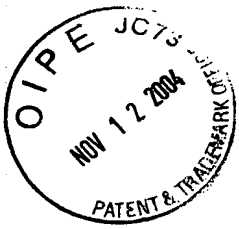
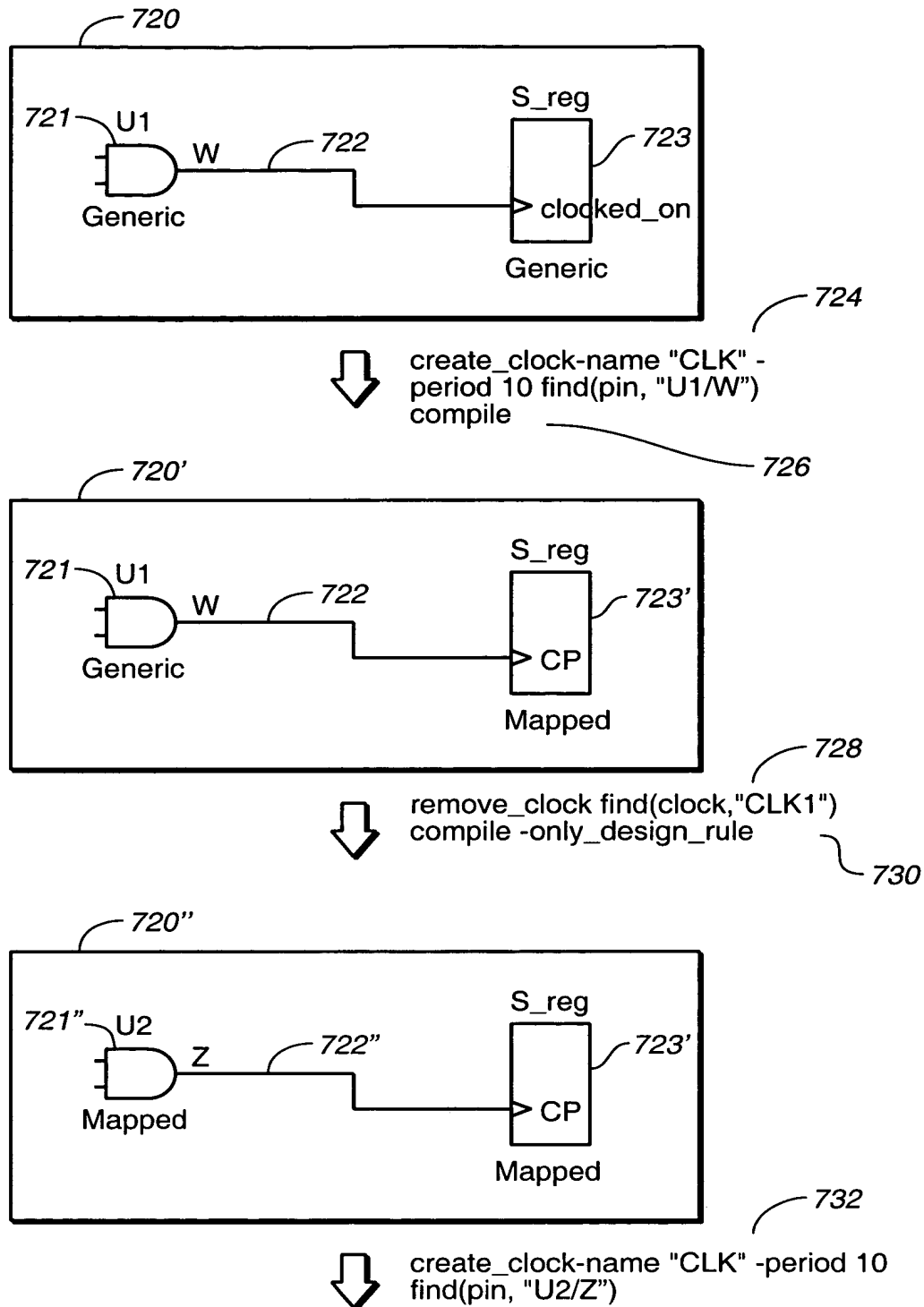
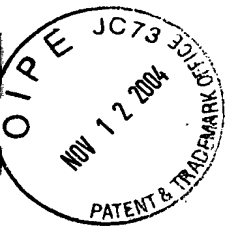


FIG._39 : External and internal clocks.

**FIG. 40**

: Process used to map cells that create internal clocks.

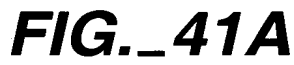
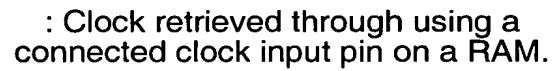


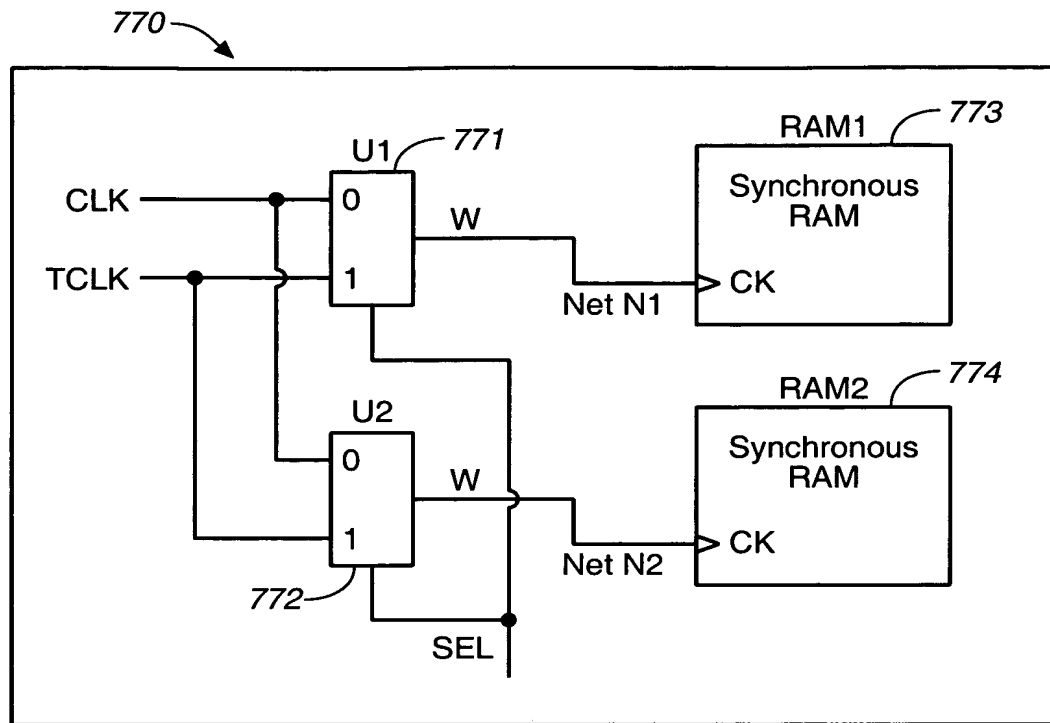
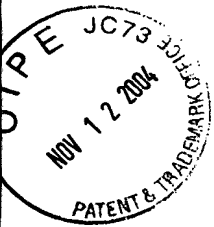
FIG._41B



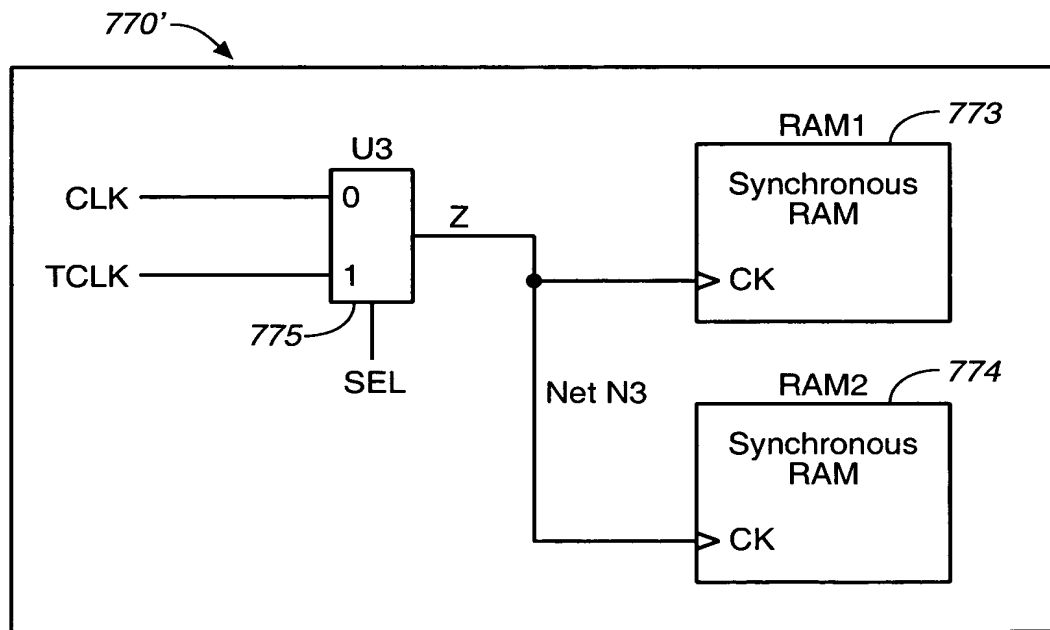
```

: Clock retrieved through using a connected net.
Retrieving names of new source pins.

```



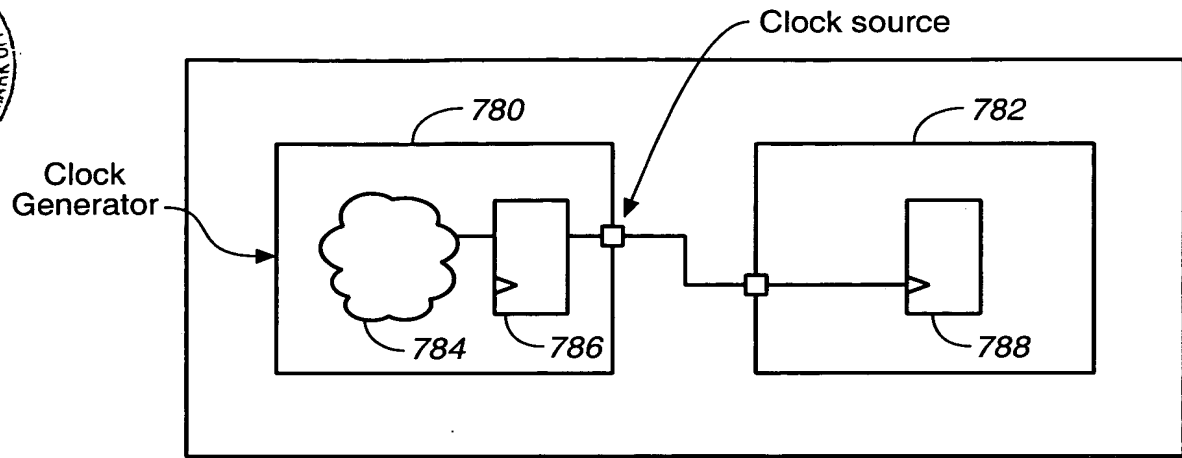
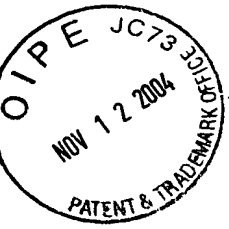
(a) Before initial mapping



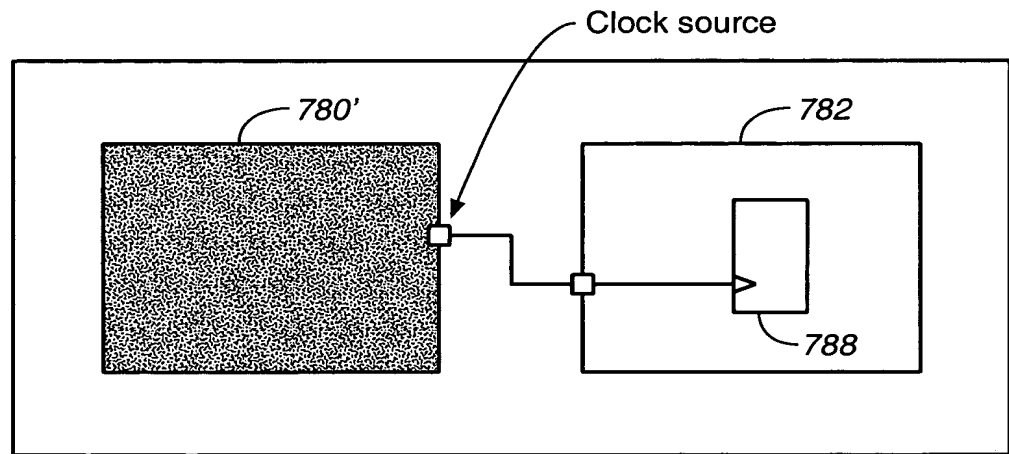
(b) After initial mapping

FIG. 42

: Example of internal clocks altered through initial mapping



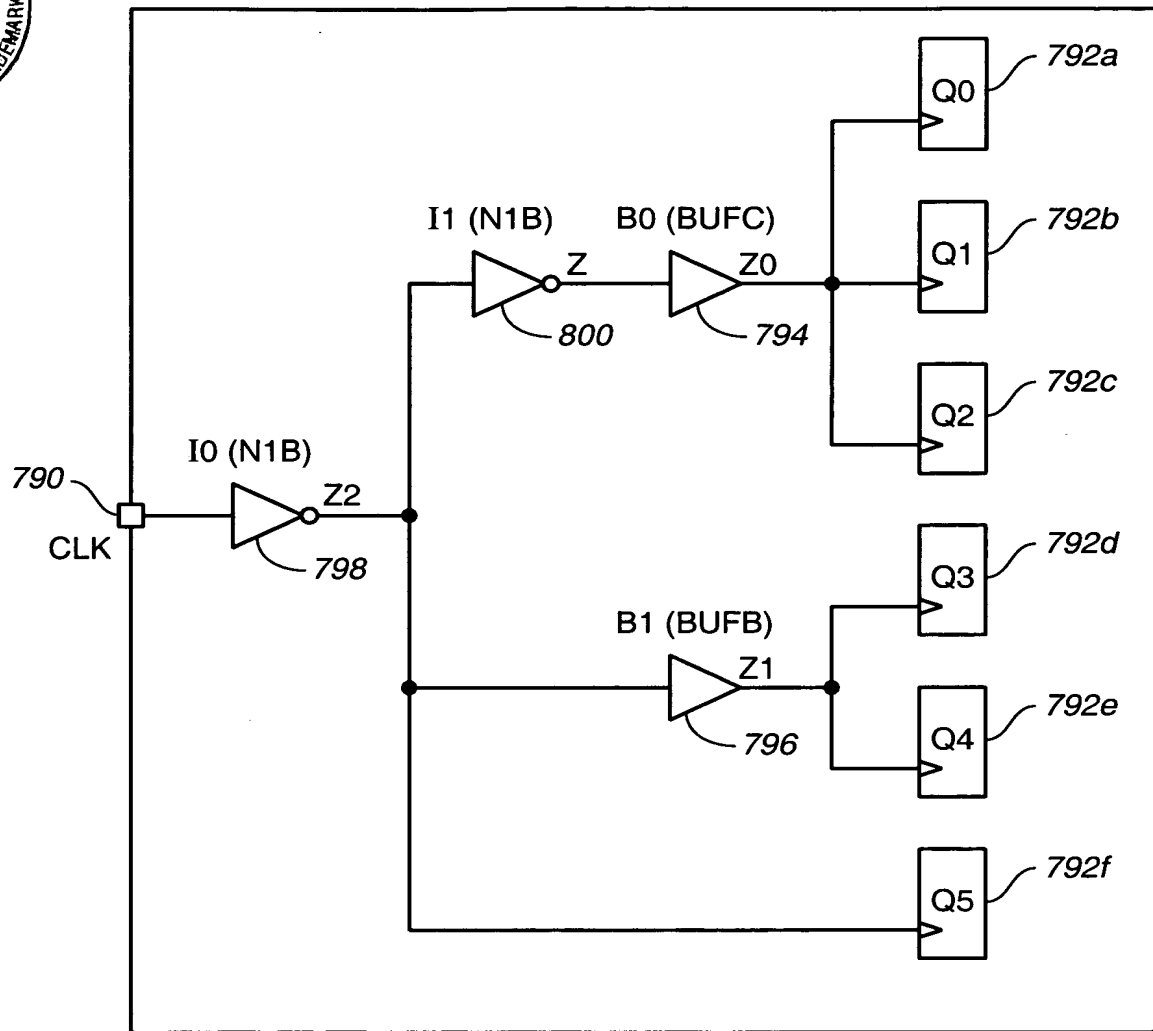
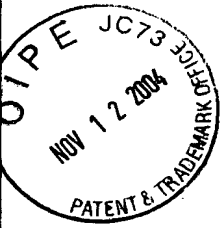
(a) Generic netlist



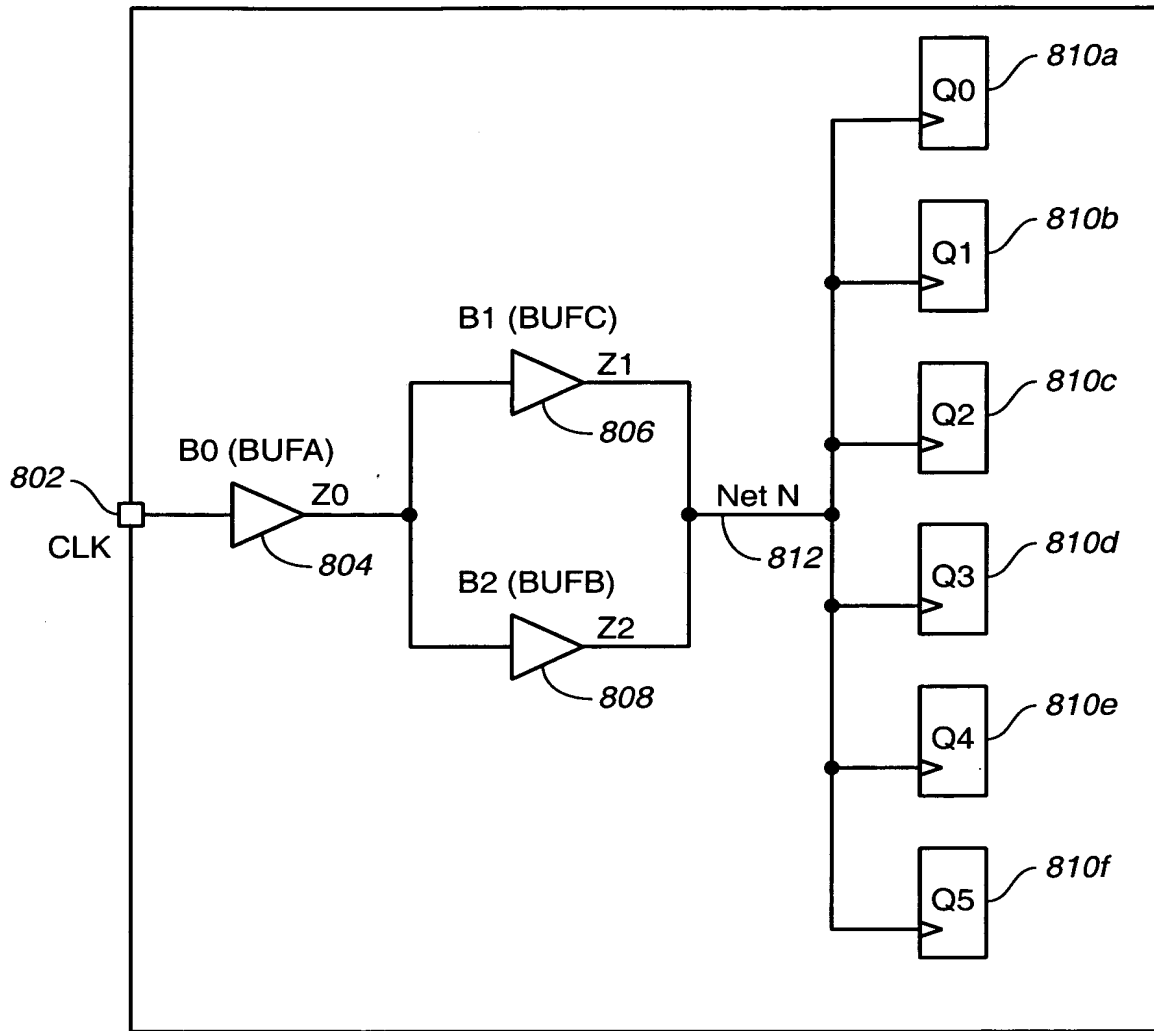
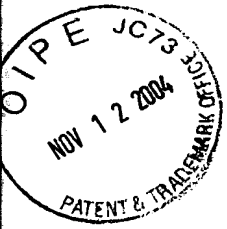
(a) After making the clock generator a blackbox for VEGA analysis

FIG._43

: Handling clock generators with a "backbox_design" directive.

**FIG. 44**

: Example of buffering tree used for clock distribution

**FIG. 45**

: Example of parallel buffers

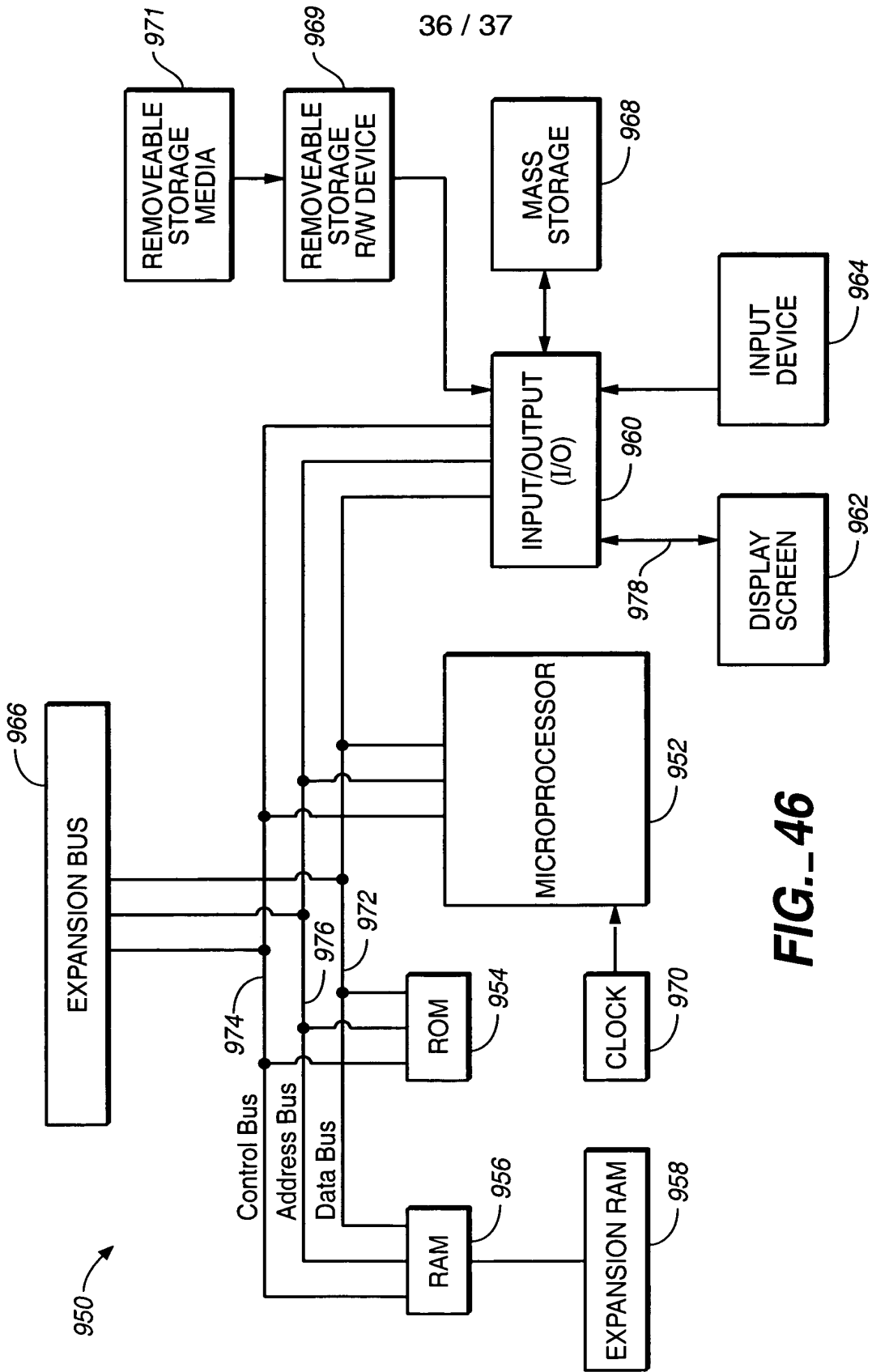


FIG. 46

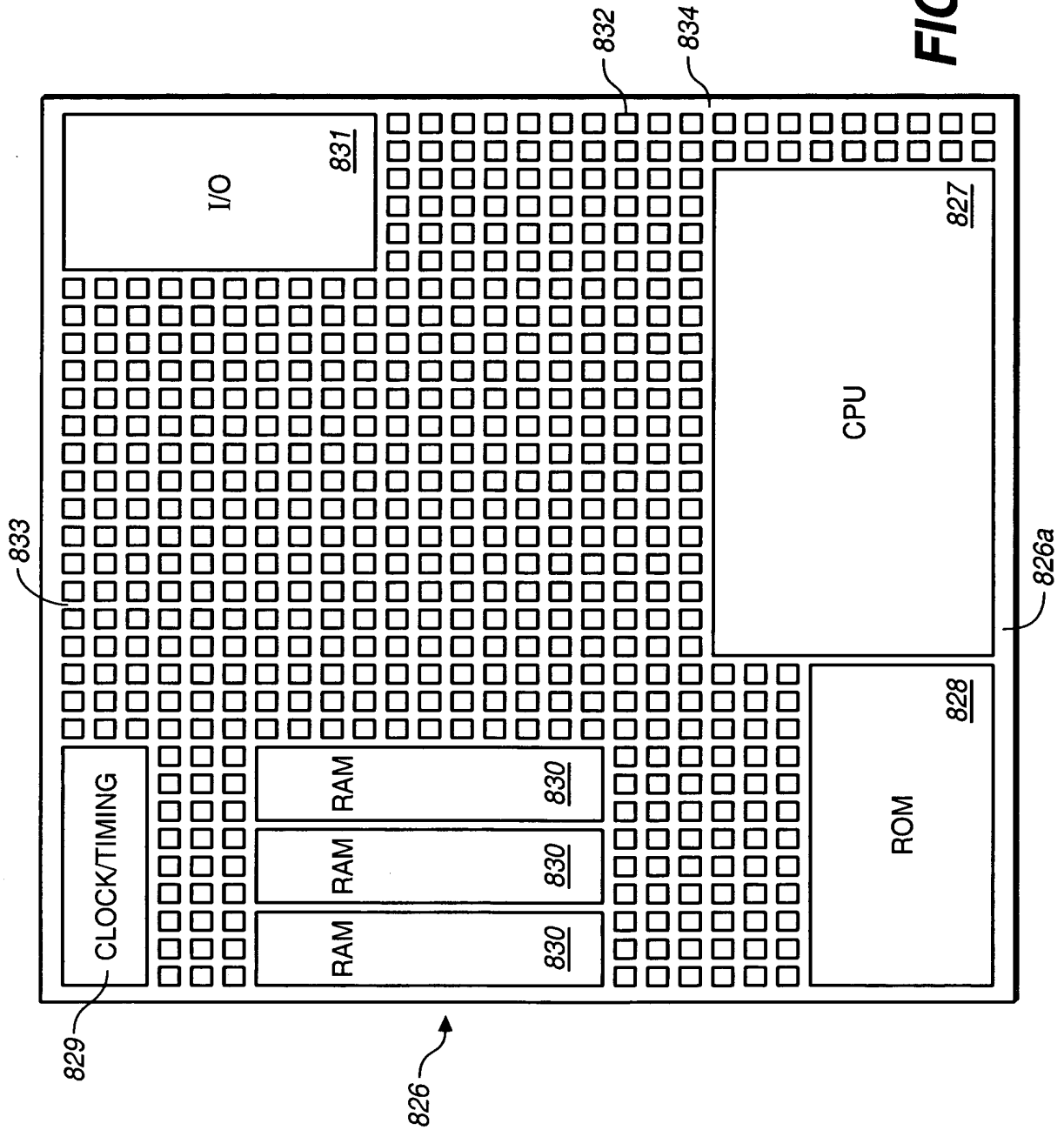


FIG. 47